DEVOPS WORKSHOP:
SPEED AND RELIABILITY

David J. Yates
Associate Professor – Bentley University
Forrest Crenshaw
Product Management Engineer – F5 Networks
DevOps Speed and Reliability Outline

- I: An Engineering Approach to DevOps (Three Ways)
- II: Agile Practices Necessary for DevOps (ACAMS+)
- III: Art & Science of DevOps (DevOps DIET)
I: An Engineering Approach to DevOps

- Agile Principles and Practices
- DevOps Principles and Practices
- Gene Kim’s Three Ways
  - First Way: The Principles of Flow
    - Continuous service delivery, systems thinking, continuous everything, …
  - Second Way: The Principles of Feedback
    - Continuous intelligence (from amplified feedback)
  - Third Way: Principles of Continual Learning
    - Continuous improvement (from experimentation, practice, adversarial thinking, …)
II: Agile Practices Necessary for DevOps (ACAMS+)

- **Agility**
  - Agile is necessary but not sufficient for DevOps

- **Focus on Culture**
  - Culture, plus people, process, and technology

- **Automation**
  - Appropriate abstraction
  - Domain-specific knowledge
  - Source of Truth
  - Imperative processes
  - Declarative interfaces
  - Orchestration & DevOps

- **Focus on Measurement**

- **Focus on Sharing**

- + Network Security
III: Art & Science of DevOps (DevOps DIET)

- Define Success
  - Set overall mission and iterative goals.

- Identify Your Path
  - Build, implement, and improve living processes with purpose.

- Execute as a Team
  - Trust, communicate, and cooperate.

- Transform and Innovate
  - Experiment, measure, decide, act.
DevOps Speed and Reliability Outline

- I: An Engineering Approach to DevOps (Three Ways)
- II: Agile Practices Necessary for DevOps (ACAMS+)
- III: Art & Science of DevOps (DevOps DIET)
I: Agile Principles

- Adaptive planning
- Evolutionary development
- Continuous improvement
  - Continuous experimentation
  - Continual learning via reflection
- Rapid and flexible response to change
I: Agile Practices

- Test-first
- Rigorous, regular refactoring
- Simple design
- Pair programming
- Sharing the work between engineers
- A single standard to which engineers adhere
- A common “war-room” style work area
I: DevOps Principles

- Continuous service delivery
  (from *Flow*, systems thinking, …)

- Continuous intelligence
  (from *Feedback*, …)

- Continuous improvement
  (from *Continual learning*, experimentation, practice, …)
I: DevOps Practices

- Configuration management
- Infrastructure as Code (IaC)
  - Revision control
  - Source of Truth
  - Change control
  - Declarative interfaces
- Continuous integration, continuous delivery
Scrum

Sprint Planning

24 hours

20 days

Sprint Backlog

Product Backlog

Sprint
Kanban

<table>
<thead>
<tr>
<th>To Do</th>
<th>Doing</th>
<th>Done</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task 4</td>
<td>Task 2</td>
<td>Task 1</td>
</tr>
<tr>
<td>Task 5</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Task 3</td>
<td></td>
</tr>
</tbody>
</table>
I: An Engineering Approach to DevOps (Review)

- Agile Principles and Practices
- DevOps Principles and Practices
- Gene Kim’s *Three Ways*
  - First Way: The Principles of Flow
    - Continuous service delivery, systems thinking, continuous everything, ...
  - Second Way: The Principles of Feedback
    - Continuous intelligence (from amplified feedback)
  - Third Way: Principles of Continual Learning
    - Continuous improvement (from experimentation, practice, adversarial thinking, ...)
DevOps Speed and Reliability Outline

- I: An Engineering Approach to DevOps (Three Ways)
- II: Agile Practices Necessary for DevOps (ACAMS+)
- III: Art & Science of DevOps (DevOps DIET)
II: Agile Practices Necessary for DevOps (ACAMS+)

- Agility
  - Agile is necessary but not sufficient for DevOps

- Focus on Culture
  - Culture, plus people, process, and technology

- Automation
  - Appropriate abstraction
  - Domain-specific knowledge
  - Source of Truth
  - Imperative processes
  - Declarative interfaces
  - Orchestration & DevOps

- Focus on Measurement

- Focus on Sharing

- + Network Security
Agility

- Agile is necessary but not sufficient for DevOps.
- DevOps is a set of practices and cultural changes — supported by the right tools — that creates an automated software delivery pipeline, enabling organizations to win, serve, and retain customers. (Source: Forrester, http://blogs.forrester.com/amy_demartine/15-06-04-what_is_devops)
- DevOps finishes what agile started. (Sources: Damon Edwards, @damonedwards; Manfred Moser, @simpligility)
Focus on Culture
- Culture, plus people, process, and technology.
- Patrick Debois, (@patrickdebois) godfather of the DevOps movement, always says DevOps is a human problem. Damon Edwards takes it one step further and claims it is a management problem.
- Deconstructing, understanding and addressing cultural challenges has evolved:

Source: IT Svit, [https://itsvit.com/blog/devops-culture-huge-step-mankind/](https://itsvit.com/blog/devops-culture-huge-step-mankind/)
Culture and PPT, Yes It’s Complicated

To achieve high velocity and high reliability, you want to develop automation for a small set of use cases which are grouped into equivalence classes (collections of use cases that are different only in minor ways). Controlling complexity in this way allows your stakeholders -- your customers, employees, and partners -- to grasp a deep understanding of the domain-specific knowledge needed to deploy, operate, and manage a limited number of classes of systems.
To achieve high velocity and high reliability, you want to develop automation for a small set of use cases which are grouped into equivalence classes (collections of use cases that are different only in minor ways). Controlling complexity in this way allows your stakeholders -- your customers, employees, and partners -- to grasp a deep understanding of the domain-specific knowledge needed to deploy, operate, and manage a limited number of classes of systems.
Automation

Source of Truth

A Source of Truth (SOT) is defined as a system or object that contains the authoritative representation of a service and its components.

When automating systems it is critical to understand that the SOT may NOT reside on the device itself, that a configuration item may be somewhere else (e.g., off-device).

As long as each configuration item is tied to one and only one SOT things work fine.

Imperative processes

An imperative process is the step-by-step set of actions required to achieve an outcome such as delivering a service.

Complexity around imperative processes can arise when you have to apply customizations, or ‘branches’, to the process which create ‘trees’ of options.

It’s the job of the expert in a particular technology, or solution, to understand which use cases can be abstracted. Thus, you can minimize the number of branches to the lowest set that delivers the service as intended.
Declarative interfaces

When we build declarative interfaces, what we really do is reduce or eliminate the need for domain-specific knowledge (all the things you need to know to complete a task).

You might ask a question of an environment and get an answer, but the answer is overwhelmingly technical. The technical stuff matters, but for the user what matters is that the interface is simple.

To create a simple interface, we make it declarative and we strive to reduce or eliminate domain-specific knowledge.
Traditionally, orchestration is implemented using imperative processes in an automated system. This increases the complexity of orchestration and requires significant domain-specific knowledge.

Perhaps orchestration needs to be done differently, to reduce the need for domain-specific knowledge. What if orchestration consumed abstracted, declarative interfaces only?

This would allow the domain-specific knowledge required for one system (e.g., your F5 BIG-IP) to be decoupled from the domain-specific knowledge required by the orchestration system (e.g., Ansible, vRo, etc.).
Agile Practices Necessary for DevOps (MS+)

- Focus on Measurement
  - You can only tell if you are innovating and improving if you are constantly measuring.
  - Measure everything related to your goals and use data to improve continuously, refine cycles, and implement change.
  - Relevant data should be collected from all teams to analyze and compare, which also fosters shared learning and cultural shifts.

- Focus on Sharing
  - Share goals, share responsibility, share knowledge, share success, and share failure.
  - Collectively own service, i.e. development, operations, all aspects of care and feeding.
  - Share learning.
  - Share change management.
  - Share unplanned work.

+ Network Security
II: Agile Practices Necessary for DevOps (Review)

- Agility
  - Agile is necessary but not sufficient for DevOps
- Focus on Culture
  - Culture, plus people, process, and technology
- Automation
  - Appropriate abstraction
  - Domain-specific knowledge
  - Source of Truth
  - Imperative processes
  - Declarative interfaces
  - Orchestration & DevOps
- Focus on Measurement
- Focus on Sharing
- + Network Security
DevOps Speed and Reliability Outline

- I: An Engineering Approach to DevOps (Three Ways)
- II: Agile Practices Necessary for DevOps (ACAMS+)
- III: Art & Science of DevOps (DevOps DIET)
III: Art & Science of DevOps

- **Requirements**
  - What does it take to succeed at DevOps?

- **Align Strategy and Tactics**
  - Bridge mission focus and teamwork
  - Bridge culture and technology

- **Iterative and Incremental Steps to DevOps Success (DIET)**
  - Define success
  - Identify your path
  - Execute as a team
  - Transform and innovate

- **Embrace Gene Kim’s Three Ways**

- **Support your evolution with ACAMS+**

- **Allow course corrections with your DevOps DIET**
What does it take to succeed at DevOps?

- Technical and cultural focus on teamwork and the service delivery mission
- Servant mindset top to bottom [Leadership] (James Hunter)
- A willingness to start where you are [Speed] (Steven Spear)
- A willingness to start with why [Reliability] (Simon Sinek)
III: Art & Science of DevOps (Strategy & Tactics)

- Align Strategy and Tactics
  - Bridge mission focus and teamwork
  - Bridge culture and technology

\[ \text{Tactics} \supseteq \text{Teamwork} \cup \text{Technology} \]
Iterative and Incremental Steps to DevOps Success

- **Step 1: Define success**
  - Set overall mission and iterative goals
- Identify your path
- Execute as a team
- Transform and innovate
Art & Science of DevOps (Step 2)

- Iterative and Incremental Steps to DevOps Success
  - Define success
  - **Step 2: Identify your path**
    - Build, implement, and improve living processes with purpose
  - Execute as a team
  - Transform and innovate
Art & Science of DevOps (Step 3)

- Iterative and Incremental Steps to DevOps Success
  - Define success
  - Identify your path
  - **Step 3: Execute as a team**
    - Trust, communicate, and cooperate
  - Transform and innovate

Diagram:
- **Define**
- **Tactics**
  - Teamwork & Technology
  - Mission Focus & Culture

- **Transform**
- **Strategy**
- **Execute**
- **Identify**
Iterative and Incremental Steps to DevOps Success

- Define success
- Identify your path
- Execute as a team
- Step 4: Transform and innovate
  - Experiment, measure, decide, act
Iterative and Incremental Steps to DevOps Success

- Define success
  - Set overall mission and iterative goals
- Identify your path
  - Build, implement, and improve living processes with purpose
- Execute as a team
  - Trust, communicate, and cooperate
- Transform and innovate
  - Experiment, measure, decide, act
Art & Science of DevOps (DIET Steps)

- Iterative and Incremental Steps to DevOps Success
  - Define success
  - Identify your path
  - Execute as a team
  - Transform and innovate
III: Art & Science of DevOps (DIET Review)

- Define success
  - Set overall mission and iterative goals

- Identify your path
  - Build, implement, and improve living processes with purpose

- Execute as a team
  - Trust, communicate, and cooperate

- Transform and innovate
  - Experiment, measure, decide, act
III: Art & Science of DevOps (Review)

- Requirements
  - What does it take to succeed at DevOps?

- Align Strategy and Tactics
  - Bridge mission focus and teamwork
  - Bridge culture and technology

- Iterative and Incremental Steps to DevOps Success (DIET)
  - Define success
  - Identify your path
  - Execute as a team
  - Transform and innovate

- Embrace Gene Kim’s Three Ways

- Support your evolution with ACAMS+

- Allow course corrections with your DevOps DIET
DevOps Speed and Reliability Outline

- I: An Engineering Approach to DevOps (Three Ways)
- II: Agile Practices Necessary for DevOps (ACAMS+)
- III: Art & Science of DevOps (DevOps DIET)
- ▶ Thank you! Other questions?