Brain Day Presentation: What is Agile and DevSecOps?

Lyndsi Hughes

David Sweeney

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Who are we?

David Sweeney

- Quick couple points talking about what we do

Lyndsi Hughes

- Systems Engineer (Operations perspective)

CERT - Engineering Focus

- Bringing **cyber to engineering** by helping organizations apply a holistic cybersecurity approach to software-intensive systems as they progress through the full system and software engineering lifecycles, defining and incorporating security strategies and automation into each phase to balance risk exposure with operational effectiveness
- Bringing **engineering to cyber** by helping organizations apply modern engineering practices to cyber domain systems and adopt emerging technologies at the rate needed to keep pace with evolving opportunities, threats, and risks

Key Expertise

Network sensing & traffic analysis

Systems supporting Security Operations

Risk Management Framework (RMF) and continuous Authority to Operate (cATO)

Model-based Systems Engineering (MBSE)

DevSecOps:
- Continuous Integration / Continuous Deployment
- Software Assurance Tool Integration
- Infrastructure-as-Code
- Containerization

Data Engineering:
- Data processing & storage at scale
- Big Data (Spark, Hadoop, Elastic Stack)
- Automated data flow (Kafka, NiFi)

Cloud Services (Amazon Web Services, Azure)

Agenda

Definitions

Benefits of Agile and DevSecOps

Real-world Examples of Agile and DevSecOps Applications

Q&A

Definitions

Working Definition of Agile

Agile:

An *iterative* and *incremental* (evolutionary) approach to software development which is performed in a *highly collaborative manner* by *self-organizing teams* within an *effective governance framework* with *"just enough" ceremony* that produces *high quality software* in a *cost effective and timely* manner which *meets the changing needs of its stakeholders*. [Ambler 2013]

[Ambler 2013] Ambler, Scott. *Disciplined Agile Software Development: Definition*. http://www.agilemodeling.com/essays/agileSoftwareDevelopment.htm

Agile Manifesto

Manifesto for Agile Software Development

February 2001

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

Individuals and interactions over processes and tools

Working software over comprehensive documentation

Customer collaboration over contract negotiation

Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

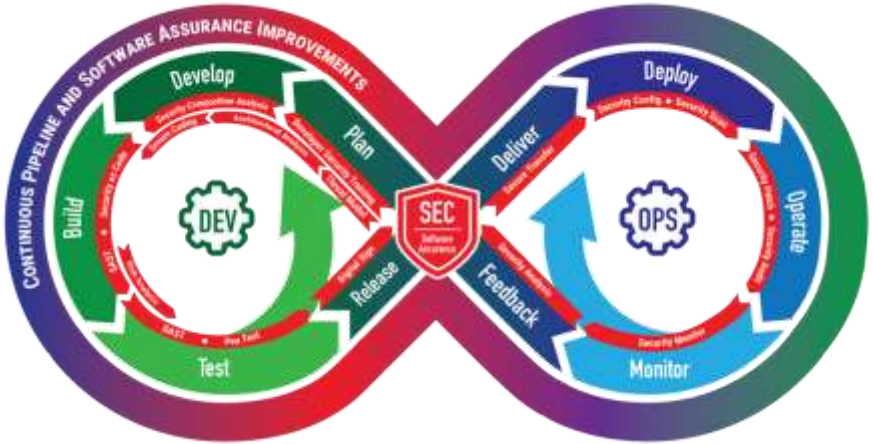
The Twelve Agile Principles[1]

1. Our highest priority is to **satisfy the customer through early and continuous delivery of valuable software**.
2. **Welcome changing requirements**, even late in development. Agile processes harness change for the customer's competitive advantage.
3. **Deliver working software frequently**, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. **Business people and developers must work together daily throughout the project.**
5. **Build projects around motivated individuals.** Give them the environment and support they need, **and trust them to get the job done**.
6. The most efficient and effective method of **conveying information** to and within a development team is **face-to-face conversation**.

The Twelve Agile Principles$_2$

7. **Working software is the primary measure of progress.**
8. Agile processes promote **sustainable development**. The sponsors, developers, and users should be able to **maintain a constant pace indefinitely**.
9. **Continuous attention to technical excellence and good design enhances agility.**
10. Simplicity—**the art of maximizing the amount of work not done**—is essential.
11. **The best architectures, requirements, and designs emerge from self-organizing teams**.
12. **At regular intervals, the team reflects on how to become more effective**, then tunes and adjusts its behavior accordingly.

DevSecOps: a Complex Socio-Technical Information System

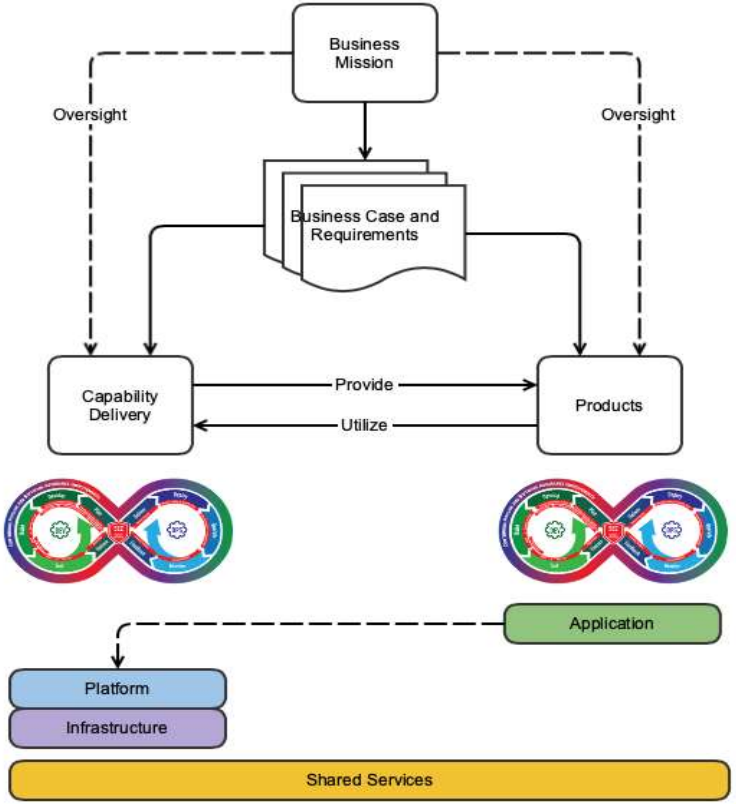


DSO is an approach that integrates development (Dev), security (Sec), and deployment/operations (Ops) of software systems to reduce the time required to move from need to capability and provide CI/CD with high software quality [1].

The DSO CI/CD pipeline is a **socio-technical system made up of both a collection of software tools and processes** [2].

It is **not a system to be built or acquired**, it is a personal and organizational **mindset** defining processes for the rapid development, fielding, and operations of software and software-based systems **utilizing automation where feasible** in order to achieve the desired throughput of new features and capabilities.

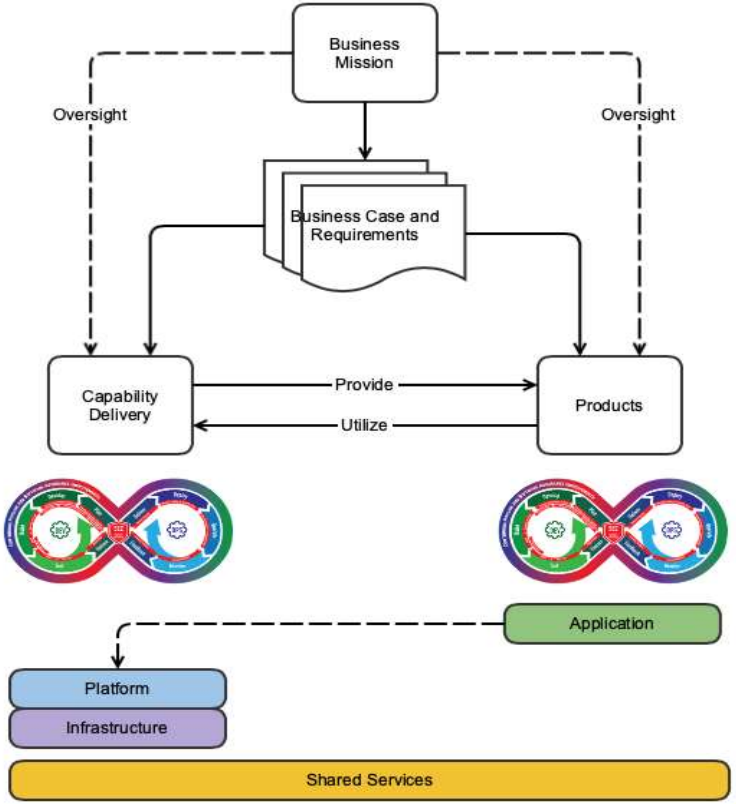
Challenge 1 for DevSecOps: connecting process, practice, & tools



Creation of the DevSecOps (DSO) pipeline for building the product is not static.

- Tools for process automation must work together and connect to the planned infrastructure
- Everything is software and all pieces must be maintained but responsibility will be shared across multiple organizations (Cloud for infrastructure, 3rd parties for tools and services, etc.)

Challenge 2 for DevSecOps: cybersecurity of pipeline and product



Managing and monitoring all of the various parts to ensure the product is built with sufficient cybersecurity and the pipeline is maintained to operate with sufficient cybersecurity is complex. Cybersecurity demands effective governance to address:

- What trust relations will be acceptable, and how will they be managed?
- What flow control and monitoring are in place to establish that the pipeline is working properly? Are these sufficient for the level of cybersecurity required?
- What compliance mandates are required? How are they addressed by the pipeline? Is this sufficient?

Benefits of Agile and DevSecOps

Benefits of Agile and DevSecOps

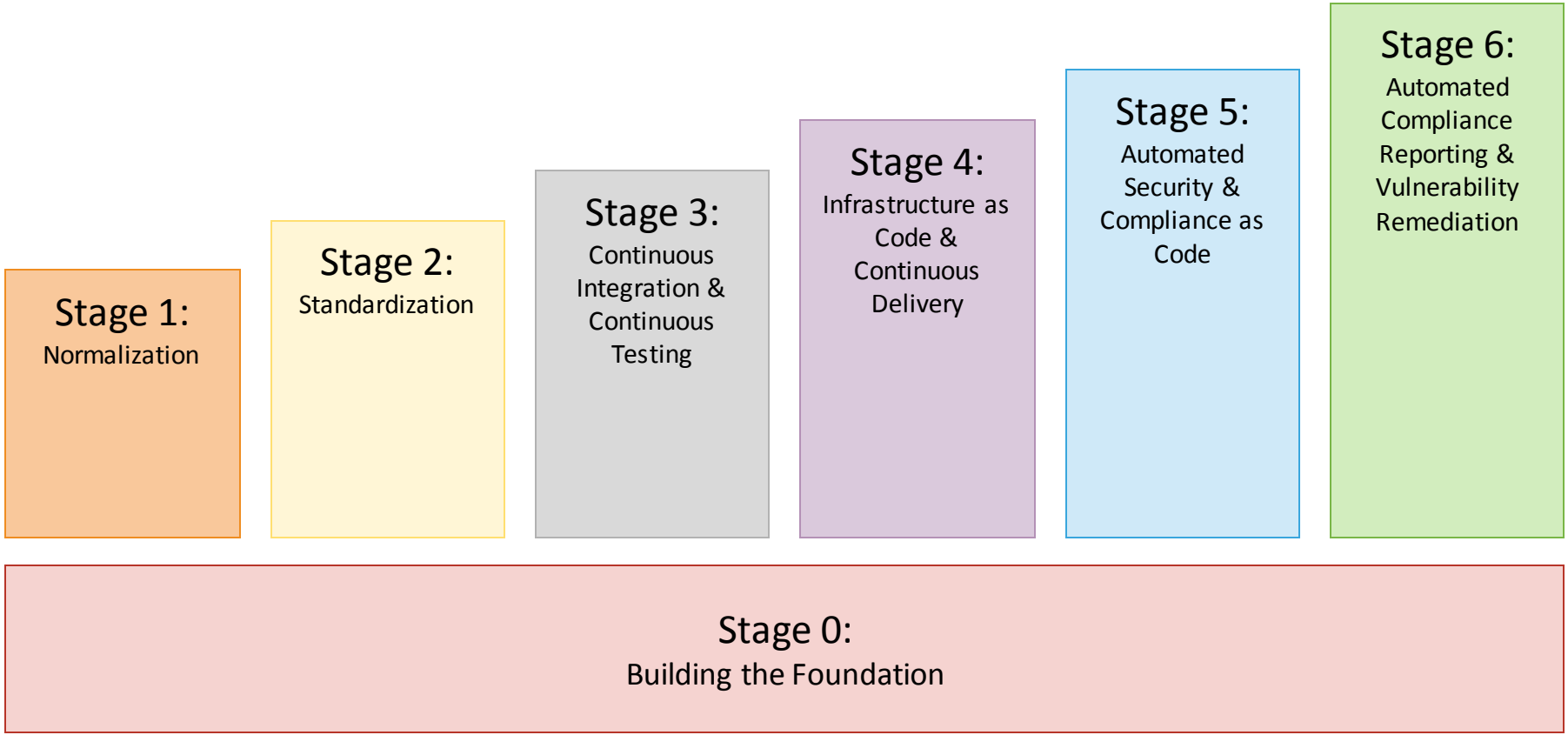
- “Just enough” ceremonies
- Iterative
- They are adaptable – you don’t have to be committed to just one!
- When used effectively, can greatly improve productivity and efficiency

Real-world Examples

Real-world Examples

1. Development Environment Operation
2. Acquisitions

Real-world Takeaways



Acquisitions engagement

First phase was to implement how the work was done in a Jira workflow

- Think of this as an MVP (Minimum Viable Product)

Second phase was to begin optimization and automation of the workflow in the workflow.

- Process improvement

Third phase was begin to examine workflow and create meaningful metrics such as average time in status etc.

- Reason for this was the workflow had to fairly stable aka not constantly having major changes in order to begin creating metrics.

Phase 1

insert original workflow

Phase 2

insert picture of halfway

Phase 3

waiting on old laptop to finish updating to pull PID workflow photos

Challenges

Challenges faced when doing a process improvement like this.

- Culture can be a massive mountain to climb.
  - The “We have always done it this way”
- It does require "buy in" from both leadership and other stakeholders.

Questions?

Contact Information

Lyndsi Hughes – lahughes@sei.cmu.edu

David Sweeney – dmsweeney@sei.cmu.edu

About the SEI - https://www.sei.cmu.edu/