Trust, Verify & Authorize with DevSecOps

Hasan Yasar

Technical Manager, Adjunct Faculty Member

Software Engineering Institute | Carnegie Mellon University

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

DM19-0184

Trust, Verify & Authorize with DevSecOps

Why DevOps?

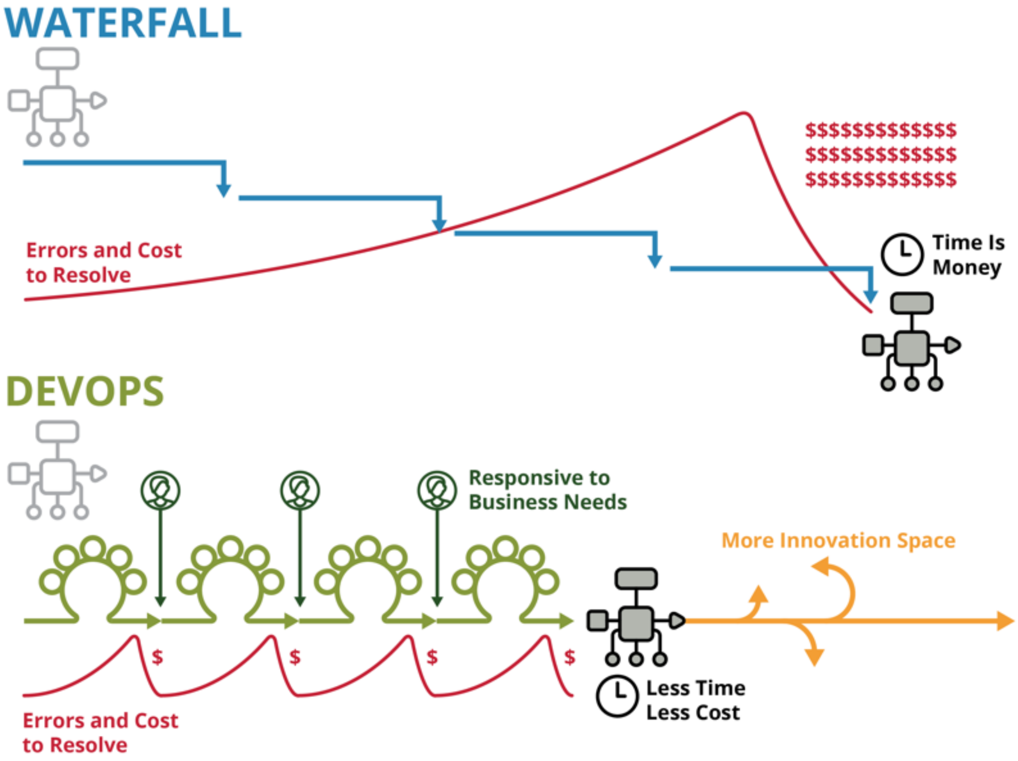
DevOps ?

DevOps is a set of principles and practices emphasizing collaboration and communication between software development teams and IT operations staff along with acquirers, suppliers, and other stakeholders in the lifecycle of a software system[1]

Four Fundamental Principles

1. *Collaboration:* between all stakeholders
2. *Infrastructure as code (IaC):* assets are versioned, scripted, and shared
3. *Automation*: deployment, testing, provisioning, any manual or human-error-prone process
4. *Monitoring*: any metric in development or operation that can inform priorities, direction, and policy

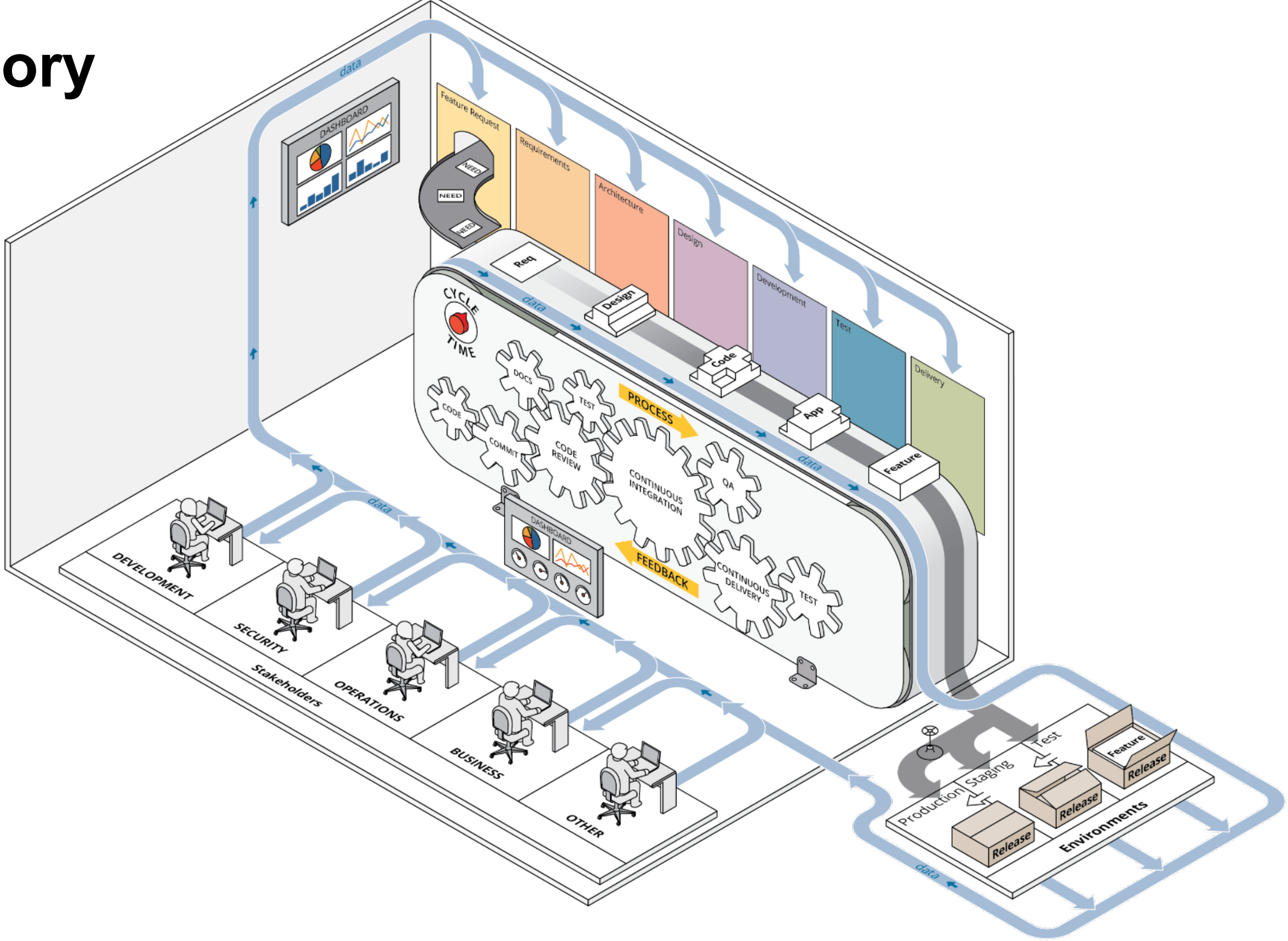
Key Benefits of DevOps



- Reduced errors during deployment
- Reduced time to deploy and resolve discovered errors
- **Repeatable** steps
- **Continuous availability** of pipeline and application
- Increased innovation time
- **Responsiveness** to business needs
- **Traceability** throughout the application lifecycle
- Increased stability and quality
- **Continuous feedback**

The DevOps Factory

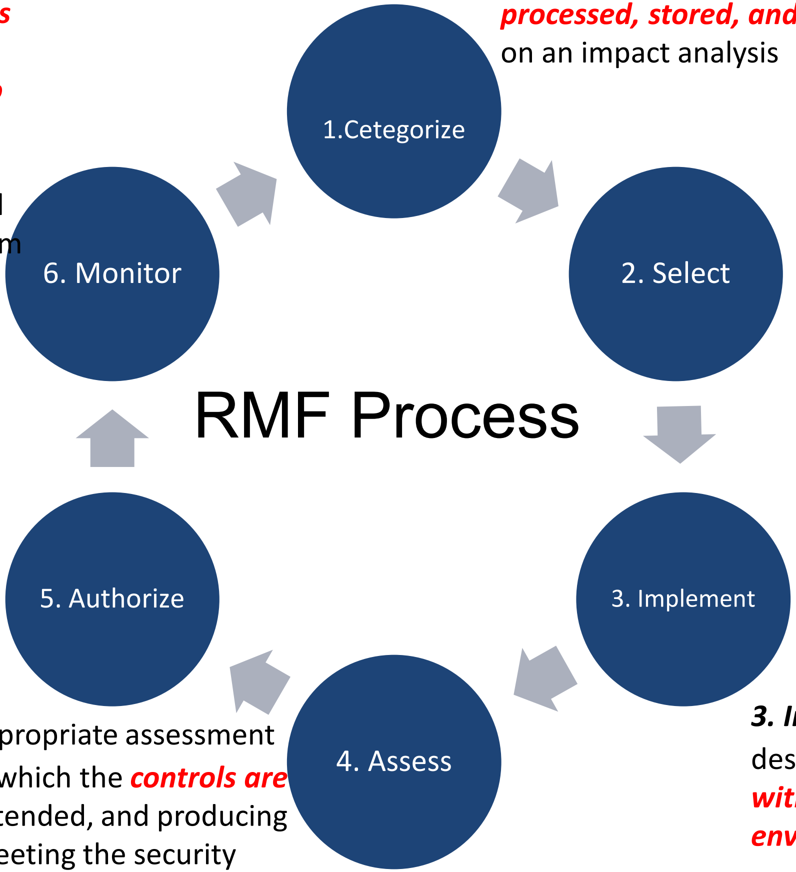
- Feature to deployment
- Iterative and incremental development
- Automation in every phase of the SDLC
- Continuous feedback
- Metrics and measurement
- Complete engagement with all stakeholders
- Transparency and traceability across the lifecycle



Trust, Verify & Authorize with DevSecOps

RMF to ATO & Compliances requirements

RMF Process



1. Categorize the information system and the ***information processed, stored, and transmitted*** by that system based on an impact analysis

2. Select an initial set of baseline security controls for the information system based on the security categorization; tailoring and supplementing the security control baseline as needed based on an ***organizational assessment of risk and local conditions***.

3. Implement the security controls and describe how ***the controls are employed within the information system and its environment of operation***.

4. Assess the security controls using appropriate assessment procedures to determine the extent to which the ***controls are implemented correctly***, operating as intended, and producing the desired outcome with respect to meeting the security requirements for the system

5. Authorize information system operation based on a ***determination*** of ***the risk to organizational operations and assets, individuals, other organizations***, and the Nation resulting from the operation of the information system and the decision that this risk is acceptable.

6. Monitor the security controls in the information system on an ***ongoing basis including assessing control effectiveness, documenting changes to the system or its environment of operation***, conducting security impact analyses of the associated changes, and reporting the security state of the system to designated organizational officials.

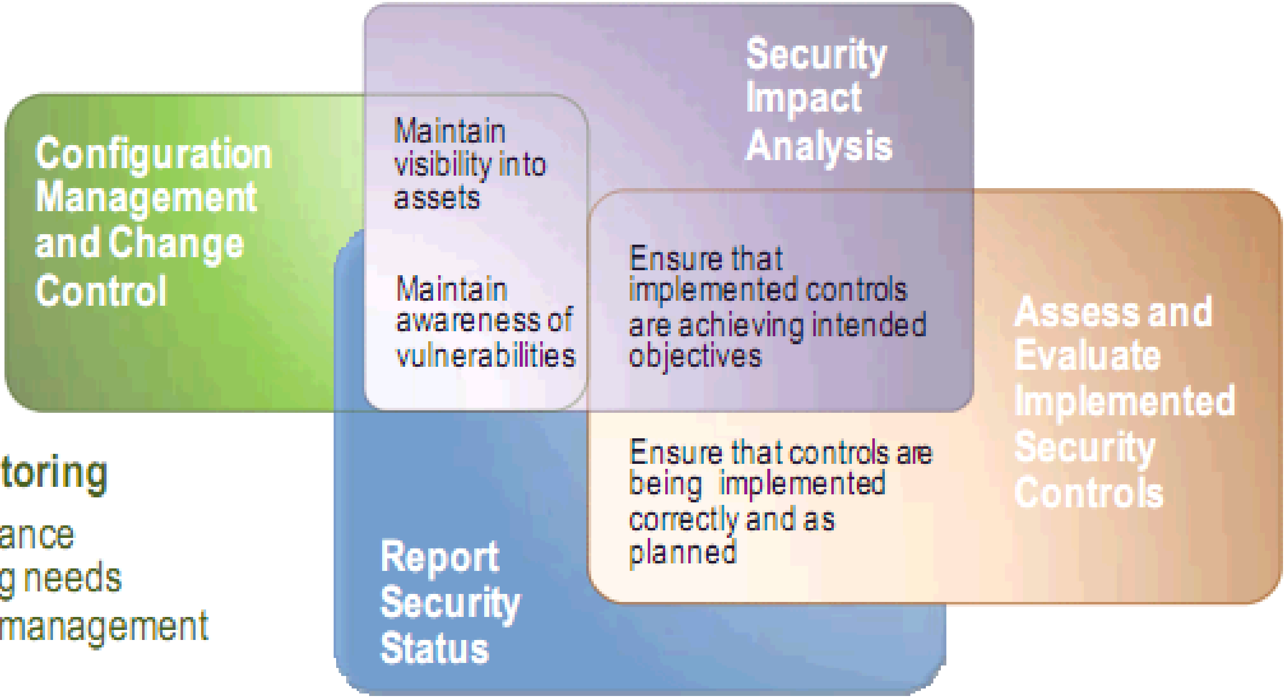
RMF characteristics – NIST 800-37

- Promotes the concept of near real-time risk management and ongoing information system authorization through the implementation of robust ***continuous monitoring processes***;
- Encourages the use of ***automation*** to provide senior leaders the necessary information to make cost-effective, risk-based decisions with regard to the organizational information systems supporting their core missions and business functions;
- Integrates information security into the enterprise architecture and ***system development life cycle***;
- Provides emphasis on the selection, implementation, assessment, and ***monitoring*** of security controls, and the authorization of information systems;
- Links risk management processes at ***the information system level*** to risk management processes at the ***organization level*** through a risk executive (function); and
- Establishes ***responsibility*** and ***accountability*** for security controls deployed within organizational information systems and inherited by those systems

Authorization with monitoring (NIST 800-137)

Continuous Monitoring

- Maps to risk tolerance
- Adapts to ongoing needs
- Actively involves management



Configuration Management and Change Control

Security Impact Analysis

Maintain visibility into assets

Maintain awareness of vulnerabilities

Ensure that implemented controls are achieving intended objectives

Assess and Evaluate Implemented Security Controls

Ensure that controls are being implemented correctly and as planned

Report Security Status

Compliance, Legal Requirements

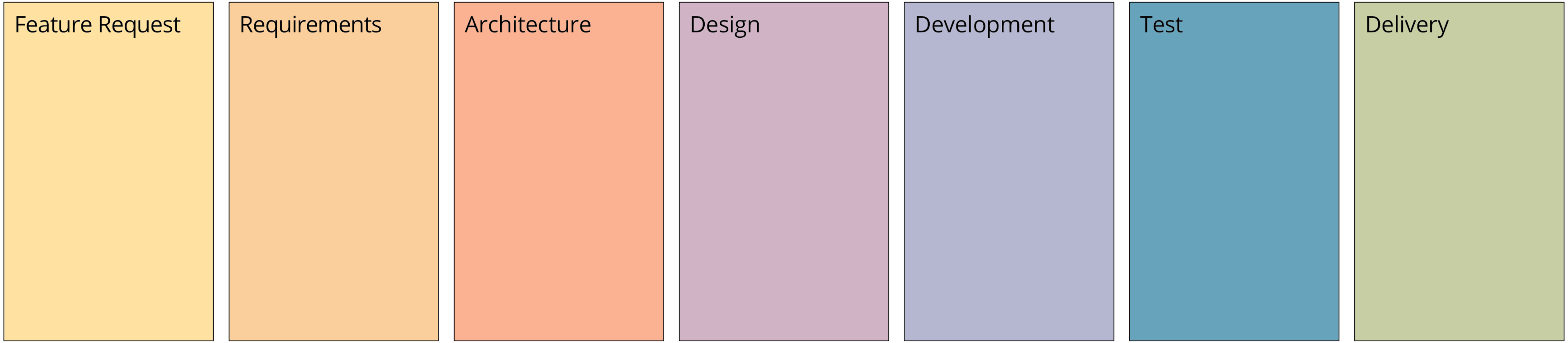
- There are many compliances and legal requirements
  - **GDPR**: General Data Protection Regulation
  - **FISMA** :Federal Information Security Management
  - **SOX** : Sarbanes–Oxley
  - **HIPAA** : Health Insurance Portability and Accountability
  - **PCI DSS**: Payment Card Industry Data Security Standard
  - **NIST** :National Institute of Standards and Technology,
  - And many more..
- All requires
  - Reporting,
  - Auditing
  - Traceability

Trust, Verify & Authorize with DevSecOps

With Secure DevOps

DevSecOps is a model on integrating the software development and operational process considering security activities: requirements, design, coding, testing , delivery , deployment and incident response.

DevOps Phases – *on each iteration/sprint*



Feature Request

Requirements

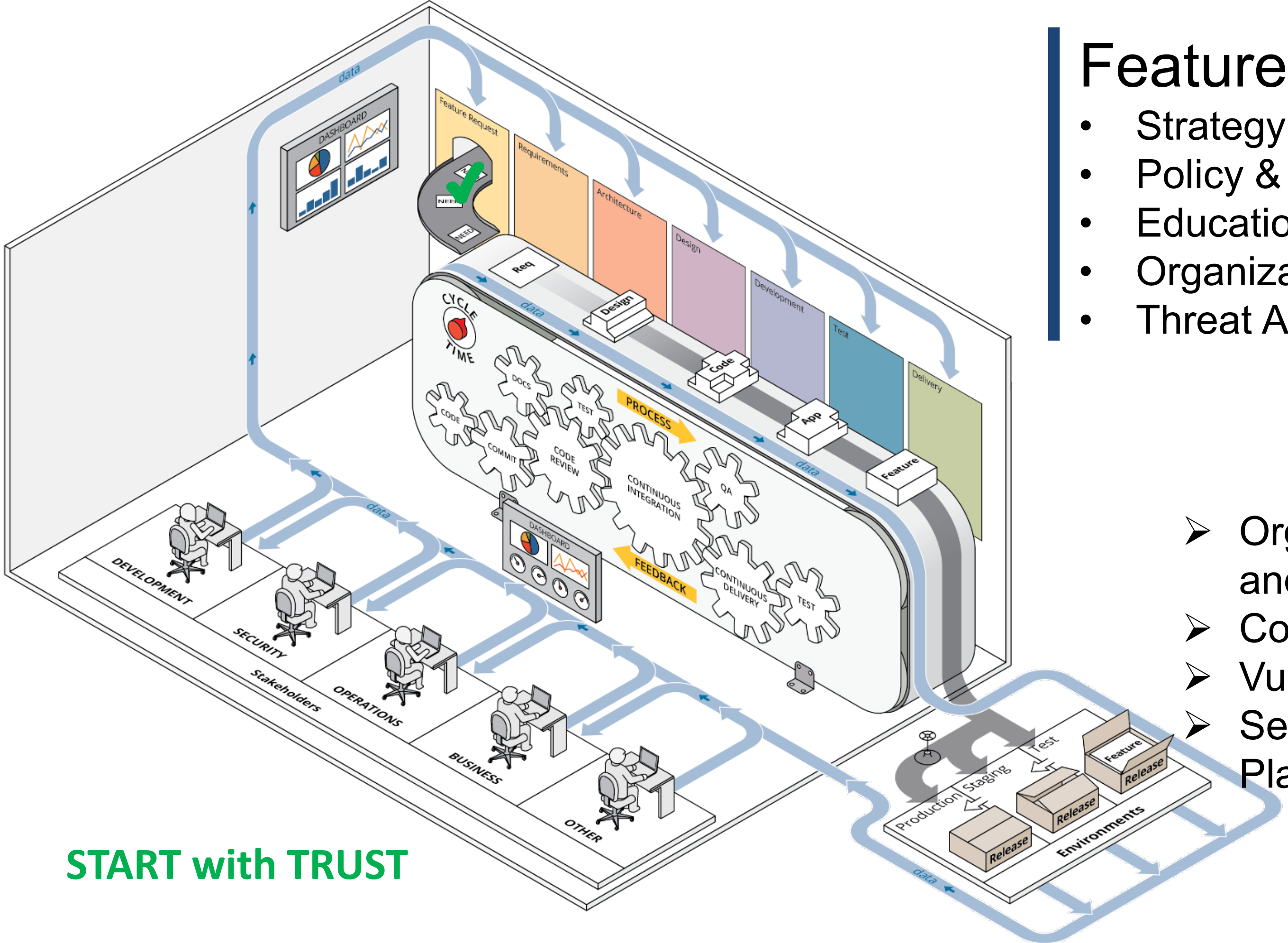
Architecture

Design

Development

Test

Delivery

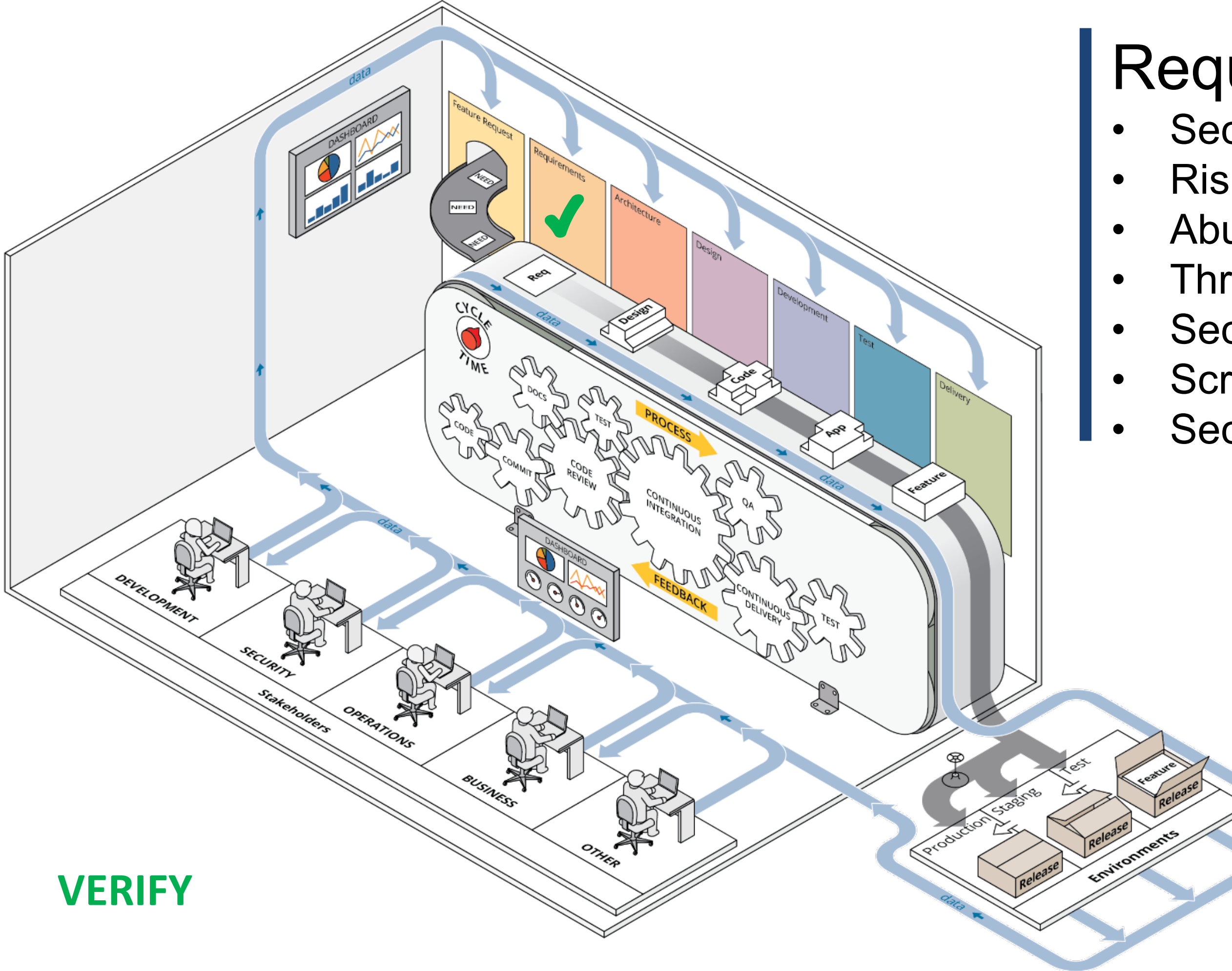


START with TRUST

Feature Request

- Strategy & Metrics
- Policy & Governance
- Education & Security Guidance
- Organizational Risk Factors
- Threat Assessment

- ➢ Organizational awareness and knowledge
- ➢ Common attack vectors
- ➢ Vulnerability management
- ➢ Security Development Plan

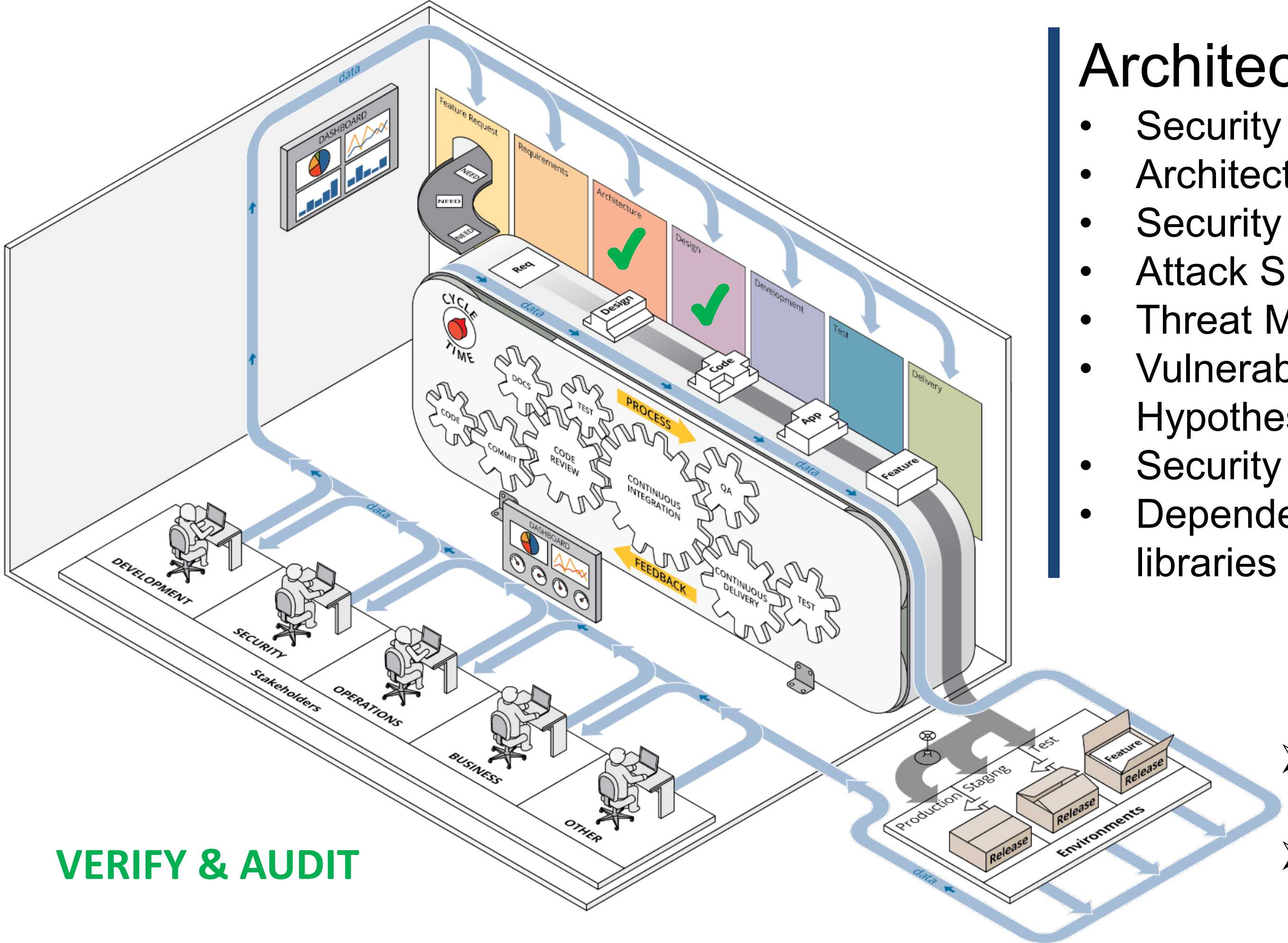


Requirements

- Security Requirements (SFR/SAR)
- Risk Assessment
- Abuse Case Development
- Threat Modelling
- Security Stories
- Screen Development Tools
- Secure/Hardened Environments

- ➢ “Baked in” Security Thoughts
- ➢ Verify Security Requirements
- ➢ Feature based security controls

VERIFY



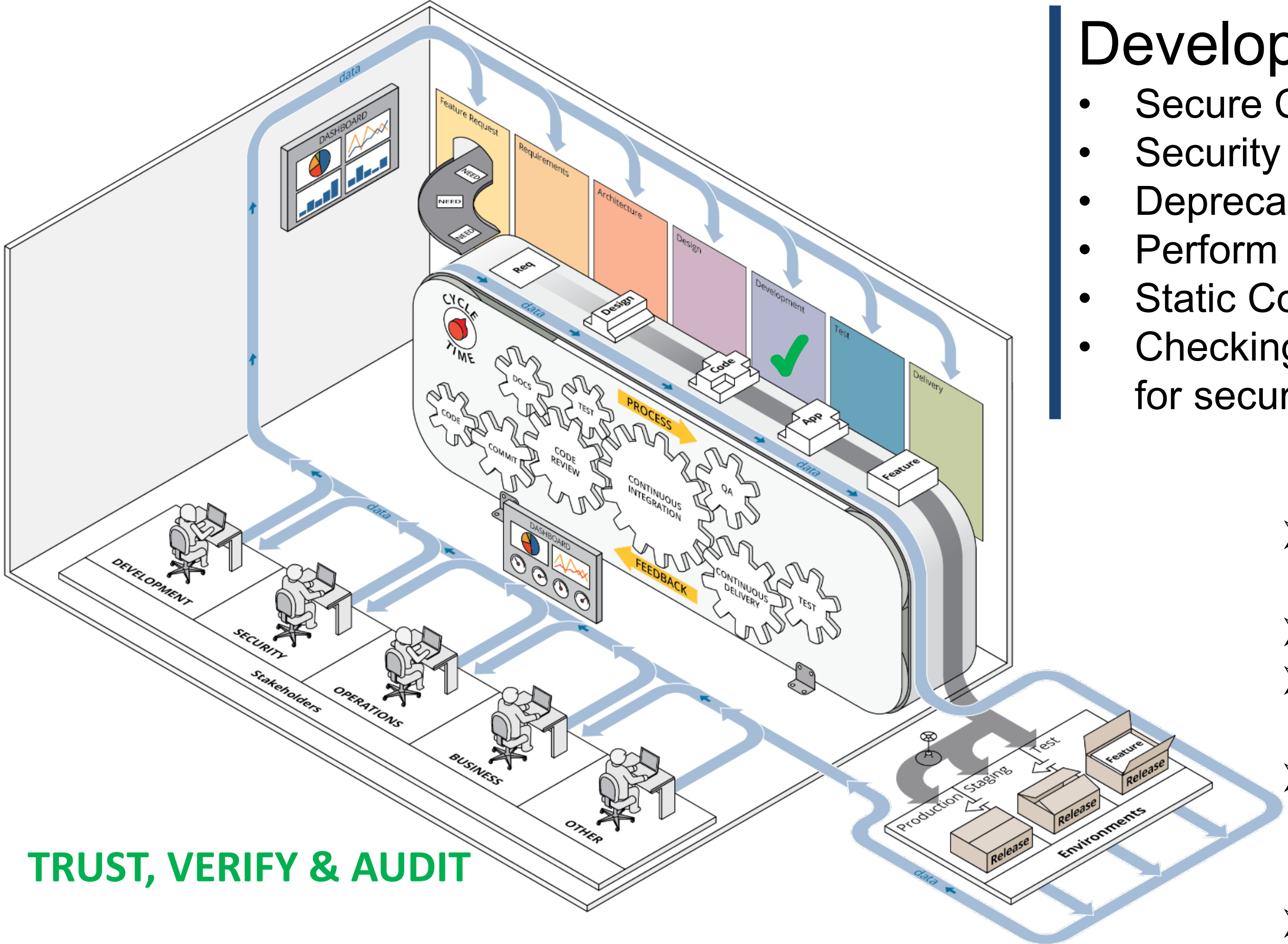
VERIFY & AUDIT

Architecture & Design

- Security Architecture
- Architectural Risk Analysis
- Security Design Requirements
- Attack Surface Analysis
- Threat Modelling
- Vulnerability Analysis and Flow Hypothesis
- Security Design Review
- Dependencies List, Open-source libraries

➢ Verify and Validate Securi Design

➢ Personnel data- privacy

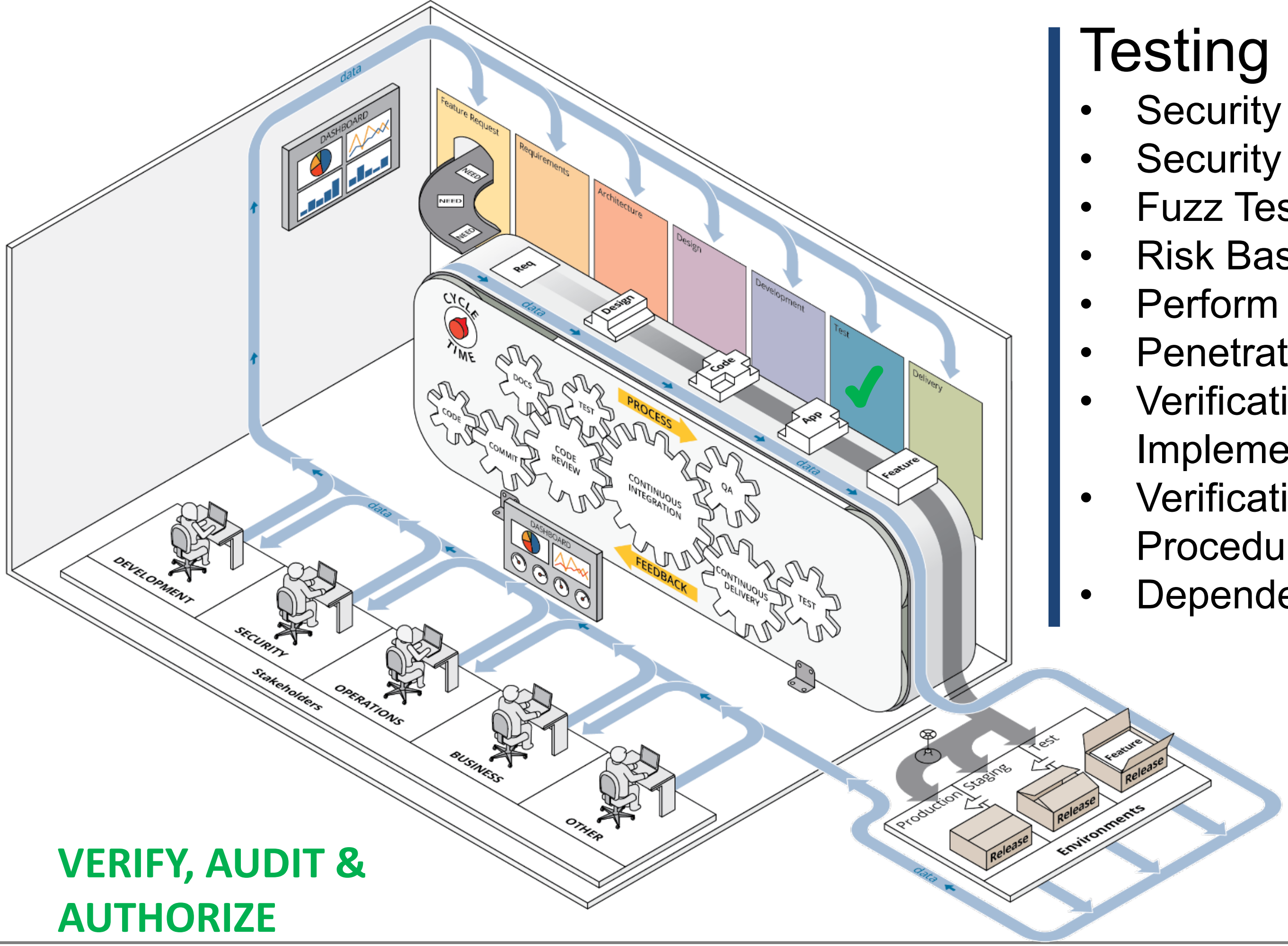


TRUST, VERIFY & AUDIT

Development

- Secure Coding Practices
- Security Focused Code Review
- Deprecate Unsafe Functions
- Perform Security Unit Testing
- Static Code Analysis
- Checking of process and procedures for secure coding & traceability

➢ Code Development Audit
➢ Unit Testing result
➢ Static Code Analysis results
➢ Code verification and validation on security practices
➢ Design validation

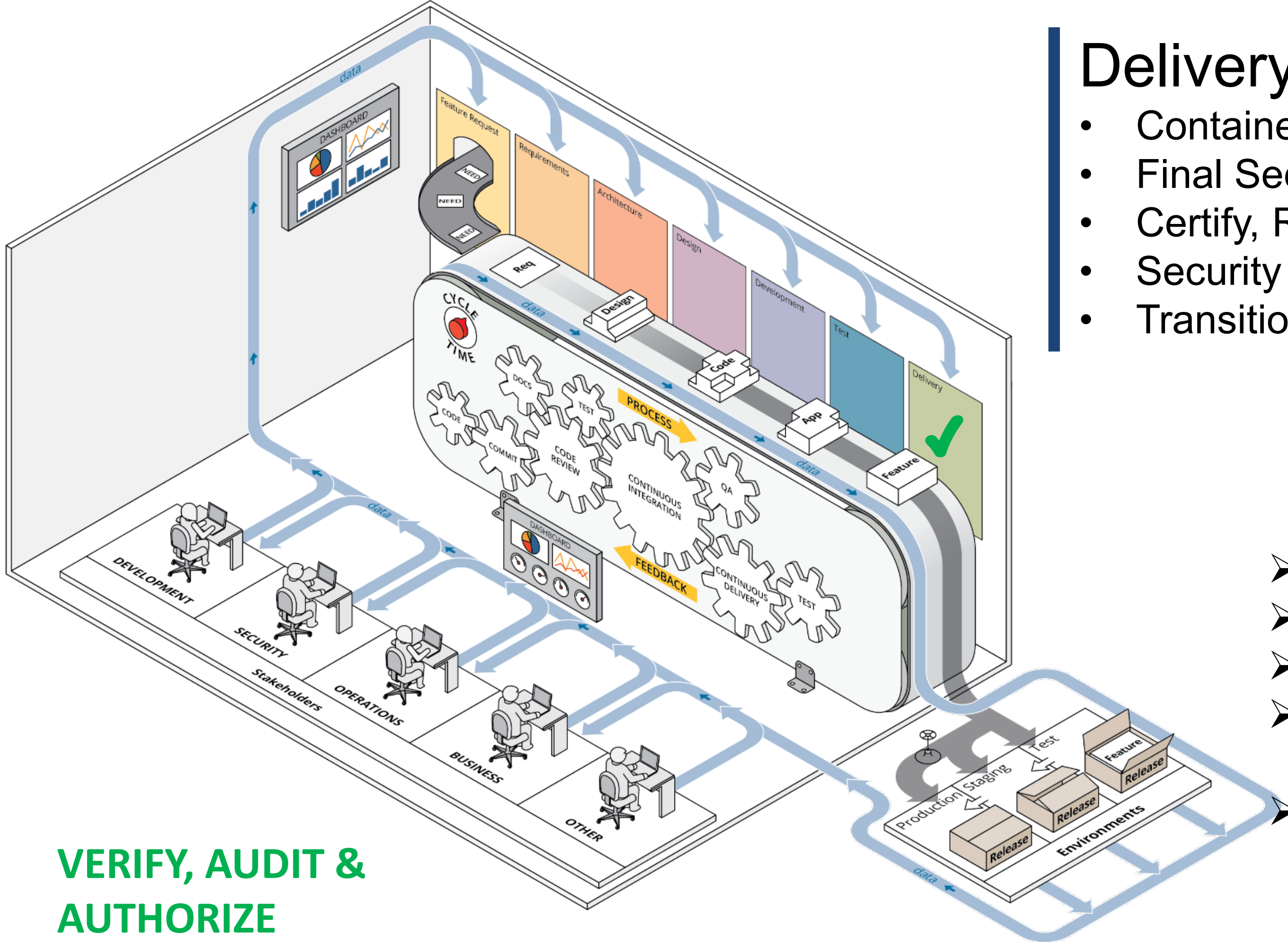


VERIFY, AUDIT & AUTHORIZE

Testing

- Security Test Planning
- Security Testing
- Fuzz Testing
- Risk Based Security Testing
- Perform Dynamic Analysis
- Penetration Testing
- Verification of Security Implementation
- Verification of Process and Procedures
- Dependency Monitoring

- ➢ Test results,
- ➢ Data handling varication
- ➢ Validation of security features

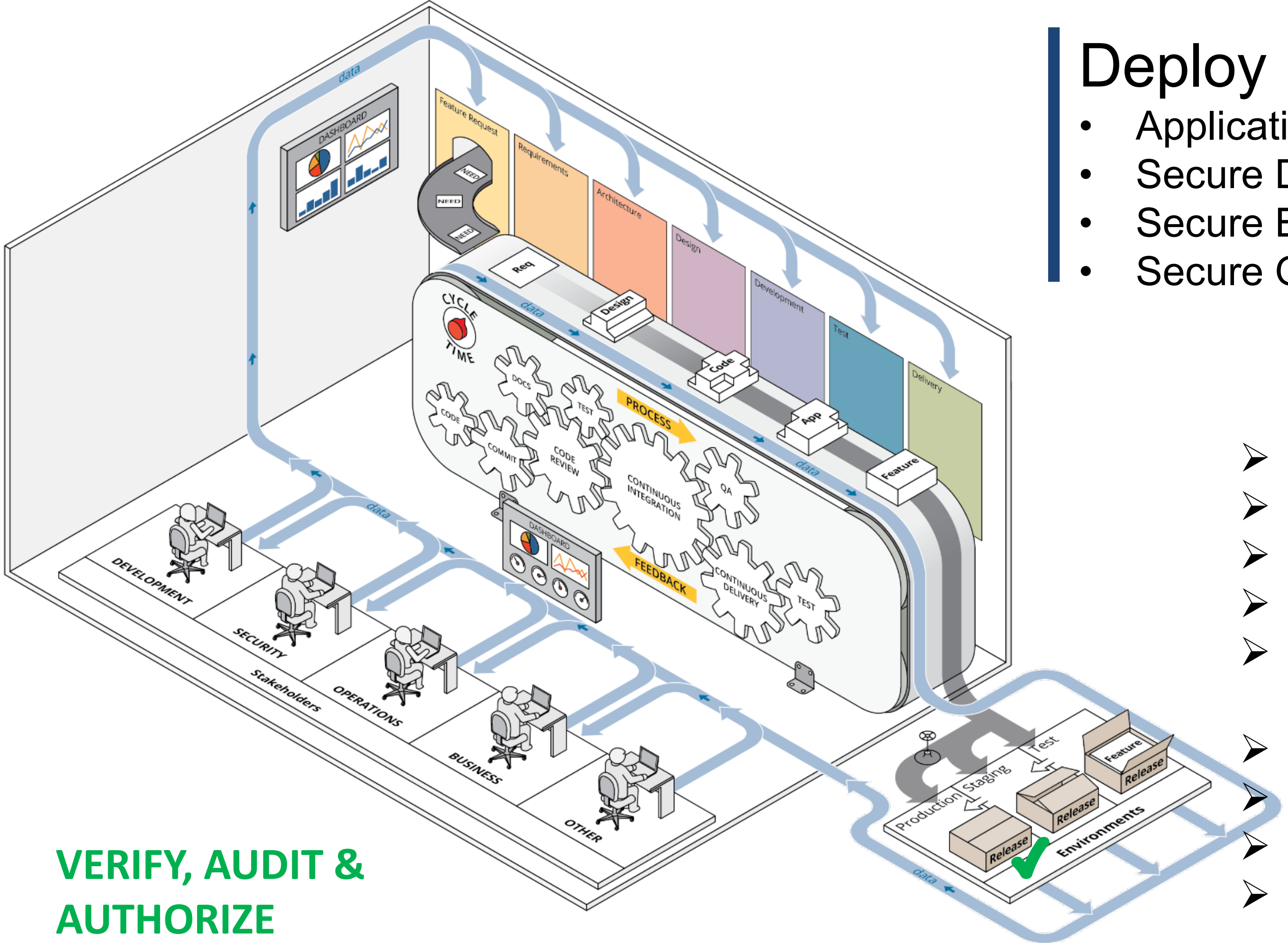


Delivery

- Container Security
- Final Security Review
- Certify, Release and Archive
- Security Acceptance Testing
- Transition Incident Response Plan

➢ Pre-approval
➢ Dependency checks
➢ Validate incident response
➢ Audit data access /rights /contents
➢ Environment verification

VERIFY, AUDIT & AUTHORIZE

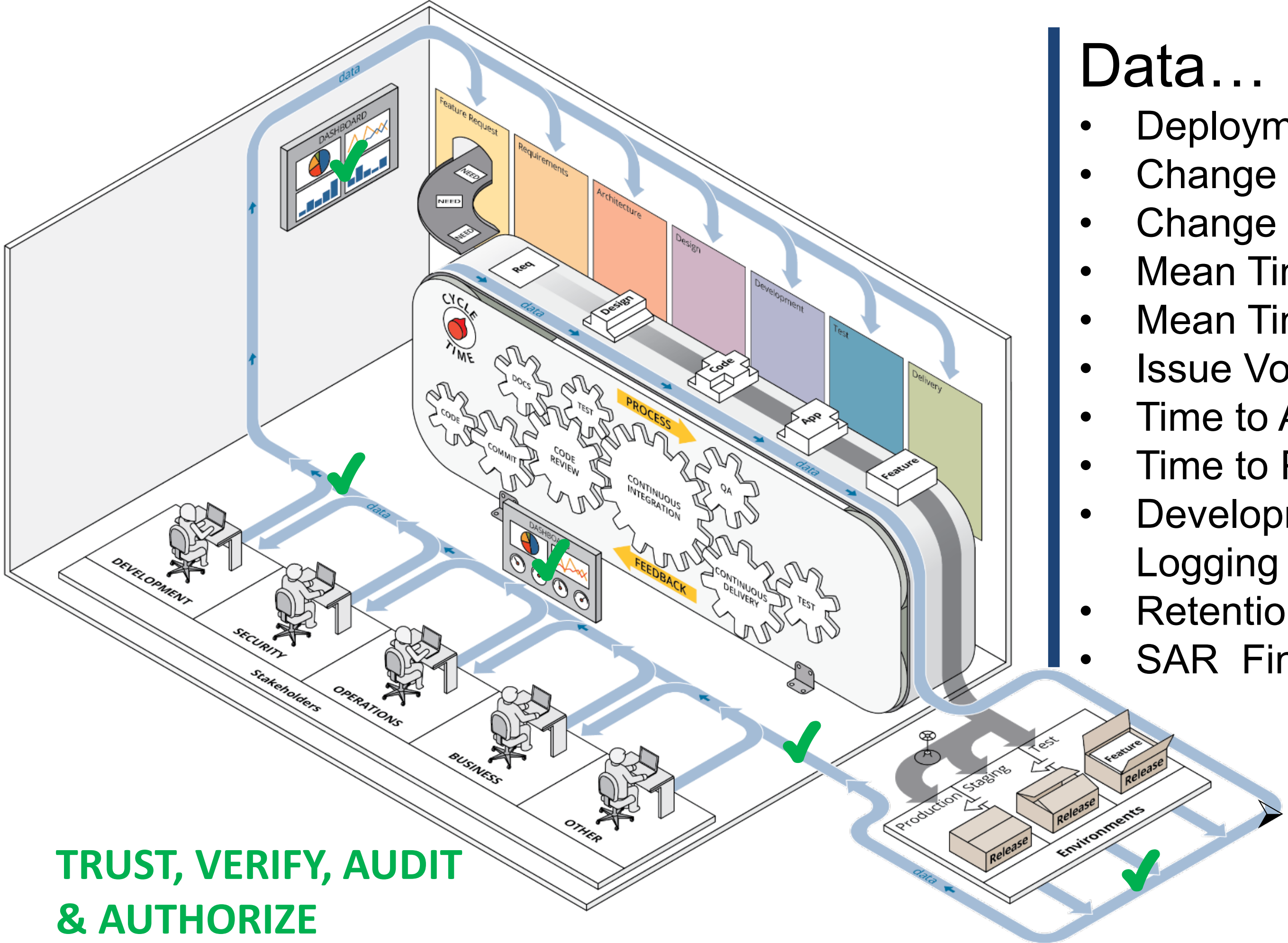


Deploy

- Application Security Monitoring
- Secure Deployment Process
- Secure Environment
- Secure Operational Enablement

- ➢ Security Dashboard
- ➢ Security Status
- ➢ Incident Response
- ➢ Rollback capabilities
- ➢ Application /Environments logs
- ➢ IDS/IPS logs
- ➢ Environment monitoring
- ➢ Resource usage
- ➢ Data handling process

VERIFY, AUDIT & AUTHORIZE

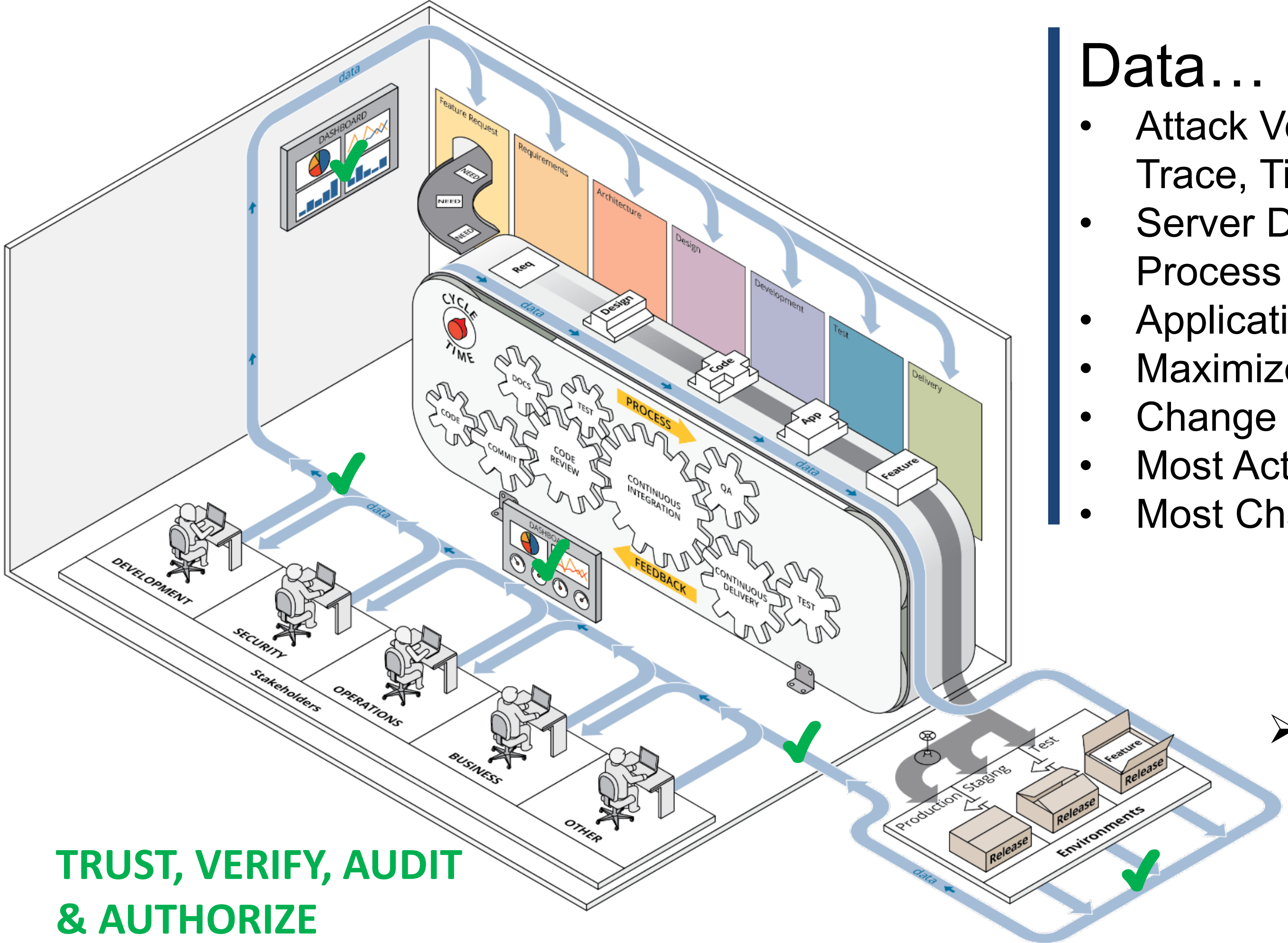


TRUST, VERIFY, AUDIT & AUTHORIZE

Data…

- Deployment Frequency
- Change Lead Time and Volume
- Change Failure Rate
- Mean Time To Recovery (MTTR)
- Mean Time to Detection (MTTD)
- Issue Volume and Resolution Time
- Time to Approval
- Time to Patch Vulnerabilities
- Development and Application Logging Availability
- Retention Control Compliance
- SAR Findings

Continuous Monitoring to feed Continuous Authorization



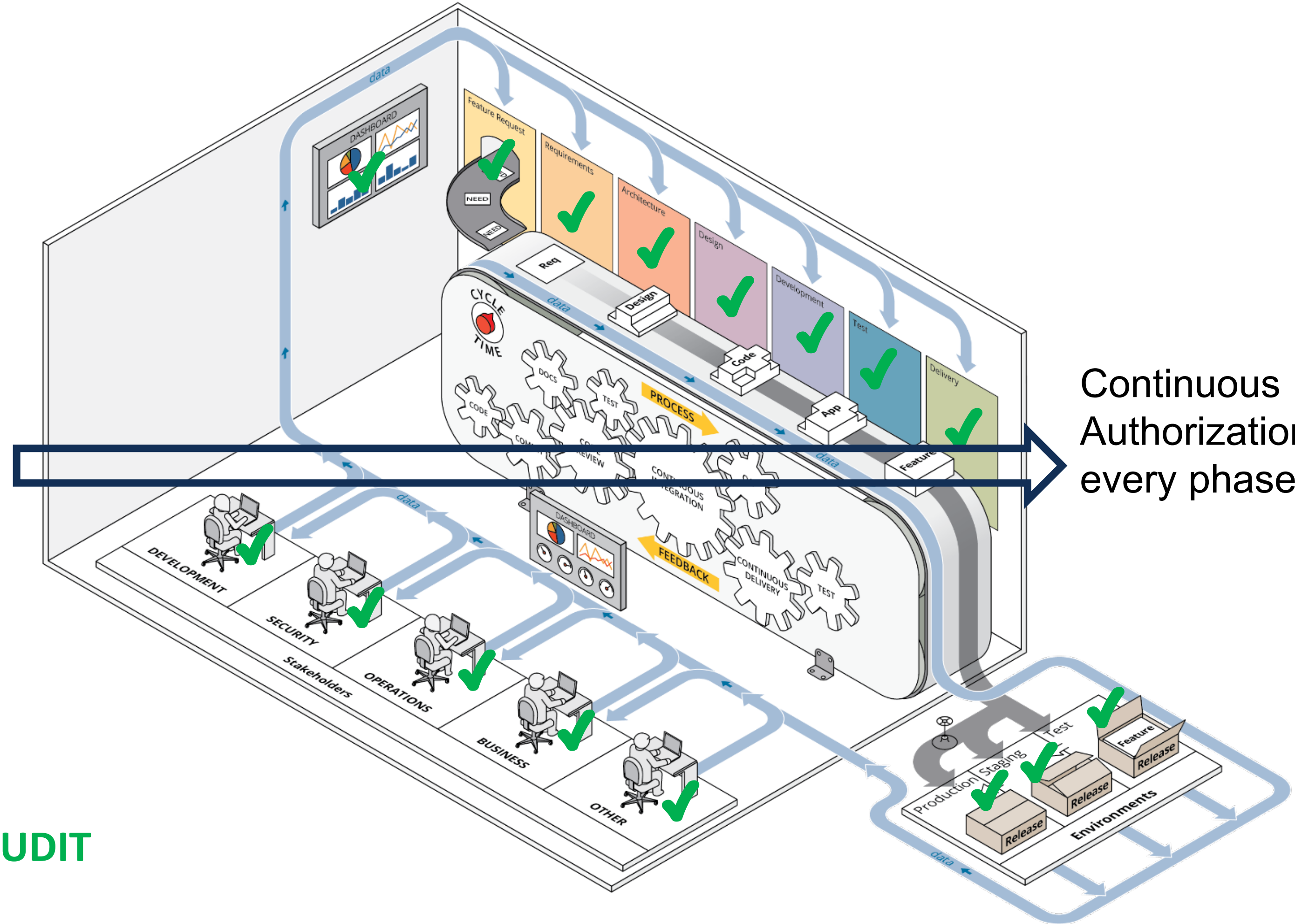
TRUST, VERIFY, AUDIT & AUTHORIZE

Data…

- Attack Vector Details (IP, Stack Trace, Time, Rate of Attack, etc)
- Server Disk Space, Load and Process Monitoring
- Application Performance
- Maximize Monitoring
- Change in Size to Code Base
- Most Active Code Contributors
- Most Changed Code Areas

➢ Continuous Monitoring to feed Continuous Authorization

Security from inception to deployment and improvement with every delivery



Continuous Authorization on every phases

TRUST, VERIFY, AUDIT & AUTHORIZE

For more information…

DevOps: https://www.sei.cmu.edu/go/devops
DevOps Blog: https://insights.sei.cmu.edu/devops
Webinars: https://www.sei.cmu.edu/publications/webinars/index.cfm
Podcasts: https://www.sei.cmu.edu/publications/podcasts/index.cfm
YouTube: https://www.youtube.com/user/TheSEICMU

SLS team GitHub Projects

- Once Click DevOps deployment
  https://github.com/SLS-ALL/devops-microcosm
- Sample app with DevOps Process
  https://github.com/SLS-ALL/flask_api_sample
  - Tagged checkpoints
    - v0.1.0: base Flask project
    - v0.2.0: Vagrant development configuration
    - v0.3.0: Test environment and Fabric deployment
    - v0.4.0: Upstart services, external configuration files
    - v0.5.0: Production environment
- On YouTube:
  https://www.youtube.com/watch?v=5nQIJ-FWA5A

Any Questions?

Hasan Yasar

Technical Manager,
Secure Lifecycle Solutions
hyasar@sei.cmu.edu
@securelifecycle