



 **SEI WEBINAR SERIES** | Keeping you informed of the latest solutions

# Carnegie Mellon University

This video and all related information and materials (“materials”) are owned by Carnegie Mellon University. These materials are provided on an “as-is” “as available” basis without any warranties and solely for your personal viewing and use.

You agree that Carnegie Mellon is not liable with respect to any materials received by you as a result of viewing the video, or using referenced websites, and/or for any consequences or the use by you of such materials.

By viewing, downloading, and/or using this video and related materials, you agree that you have read and agree to our terms of use ([www.sei.cmu.edu/legal/](http://www.sei.cmu.edu/legal/)).

Distribution Statement A: Approved for Public Release; Distribution is Unlimited

© 2016 Carnegie Mellon University.

Copyright 2016 Carnegie Mellon University

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Department of Defense.

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.

[Distribution Statement A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at [permission@sei.cmu.edu](mailto:permission@sei.cmu.edu).

CERT® is a registered mark of Carnegie Mellon University.

DM-0003515

# Continuous Integration Secure DevOps

Hasan Yasar,

Technical Manager

Secure Lifecycle Solutions

Software Engineering Institute  
Carnegie Mellon University  
Pittsburgh, PA 15213



Software Engineering Institute

Carnegie Mellon University

#SEIwebinar

© 2016 Carnegie Mellon University

[Distribution Statement A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

# What Is DevOps?

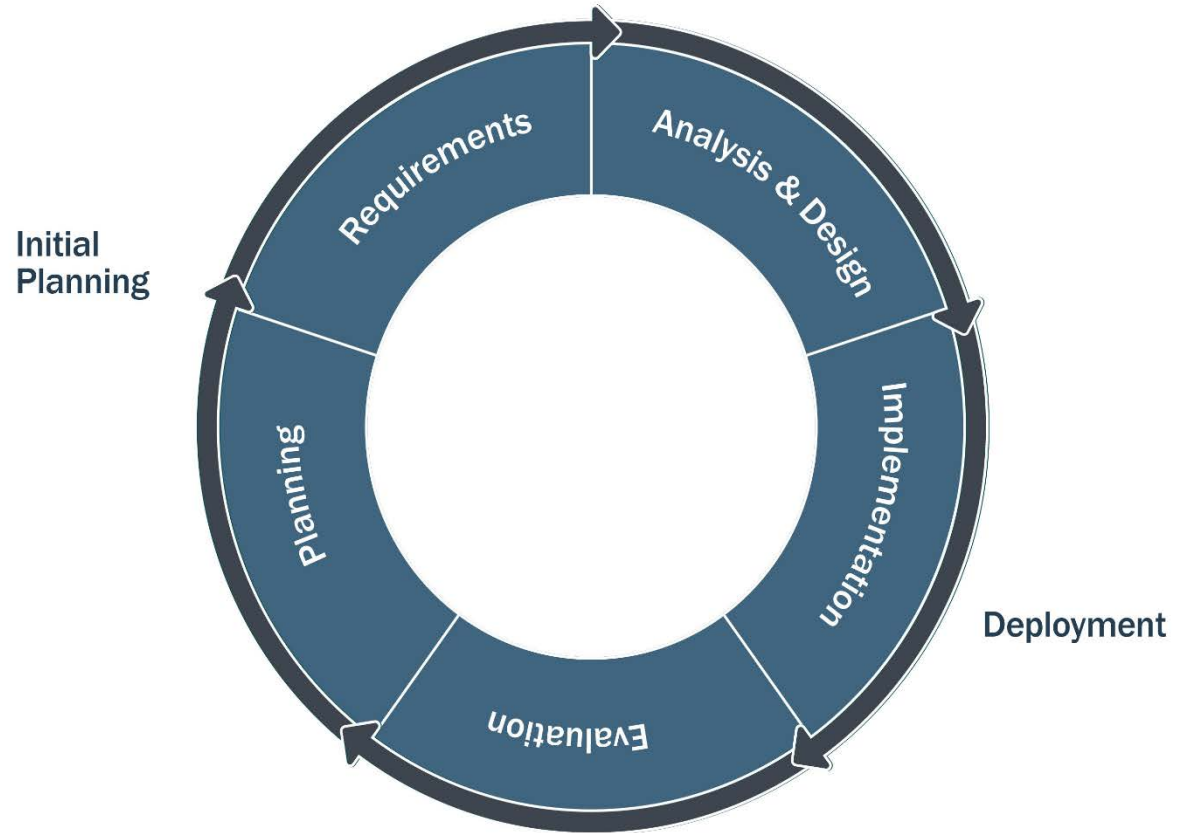


# The DevOps Movement Began as a Reaction ...



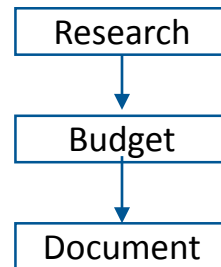
to years of disconnect between Development and Operations that began to manifest itself as conflict and inefficiency

Agile Method

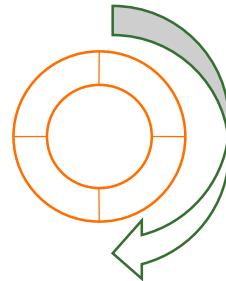


# Water - Scrum - Fall

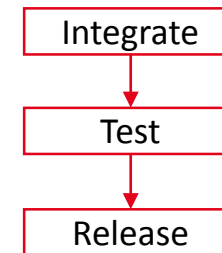
Business



Development



QA  
Operations

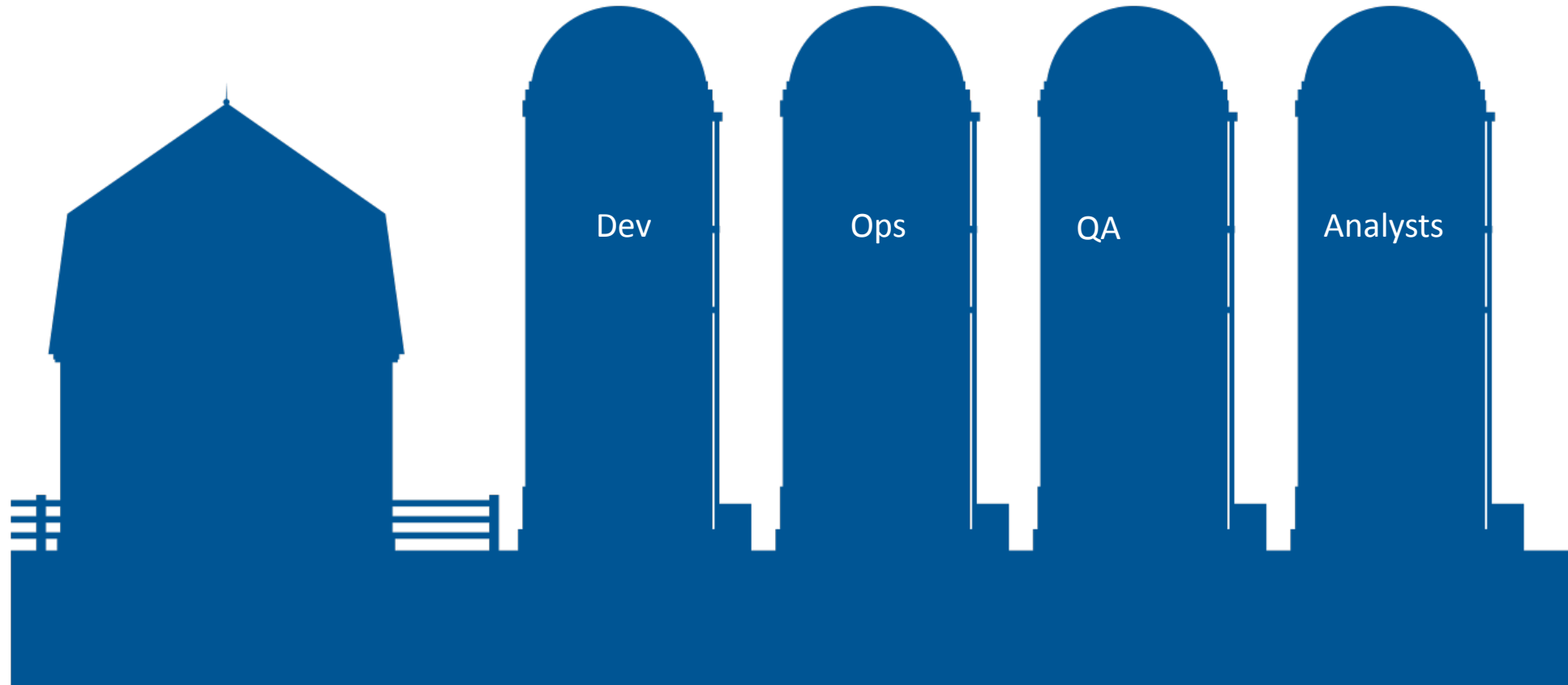


Jez Humble, [https://youtu.be/L1w2\\_AY82WY](https://youtu.be/L1w2_AY82WY)

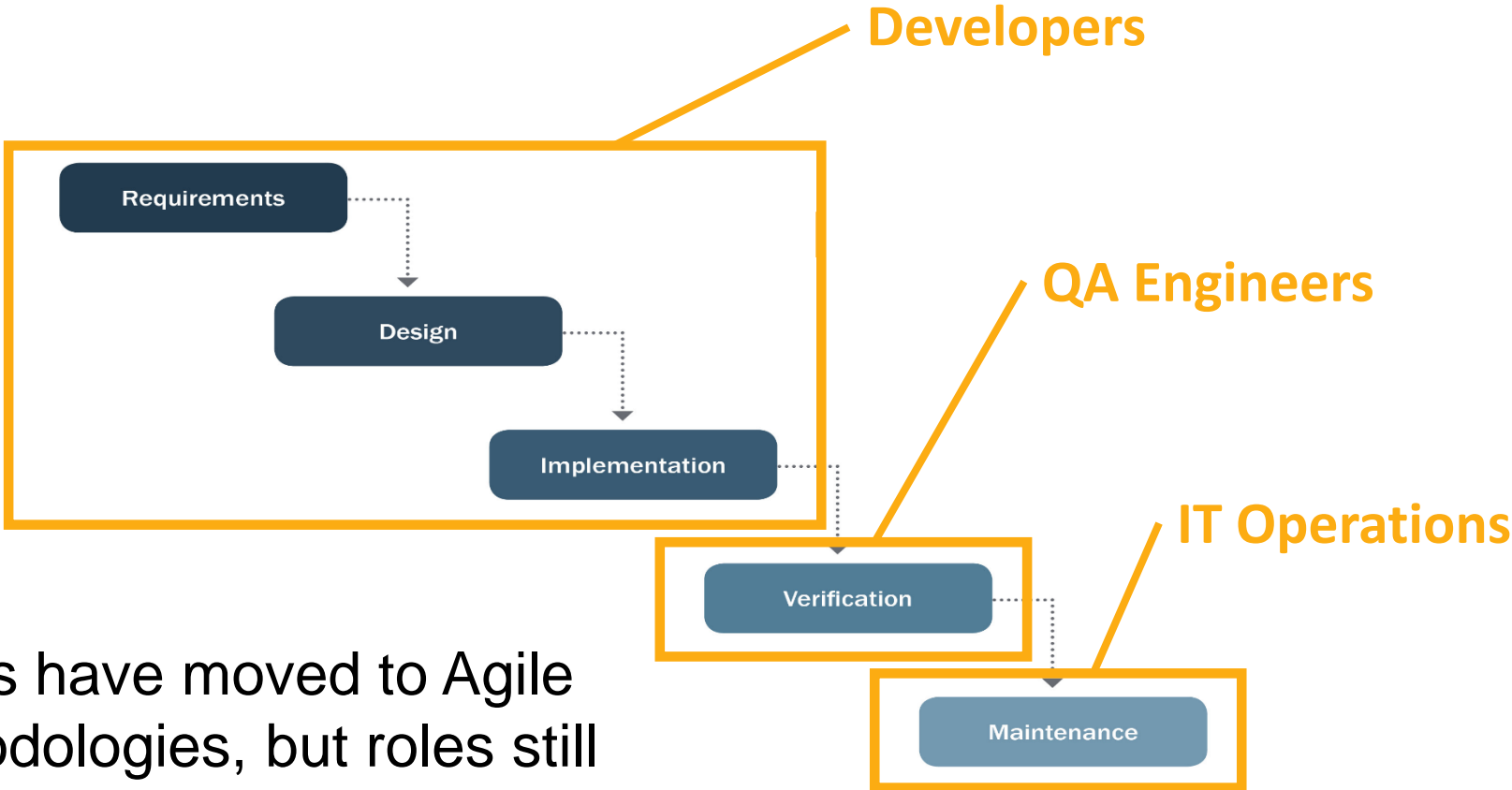
Dave West, <http://sdtimes.com/analyst-watch-water-scrum-fall-is-the-reality-of-agile/>



# Silos Block Collaboration



# Silos Reinforce Waterfall



Teams have moved to Agile methodologies, but roles still align with waterfall methods

# Polling Question

Would you like more information about DevOps?

1. Yes

2. No

# DevOps is an Extension of Agile Thinking

## Agile

**Embrace** constant change

**Embed Customer** in team to internalize expertise on requirements and domain

## DevOps

**Embrace** constant testing, delivery

**Embed Operations** in team to internalize expertise on deployment and maintenance

# DevOps Aims to Increase...

...the pace of **innovation**

...**responsiveness** to business needs

...**collaboration**

...software **quality**

# DevOps Has Four Primary Focus Areas

Collaboration between project team roles

Infrastructure as Code: Scripted Infrastructure Configuration

Automation of Tasks / Processes / Workflows

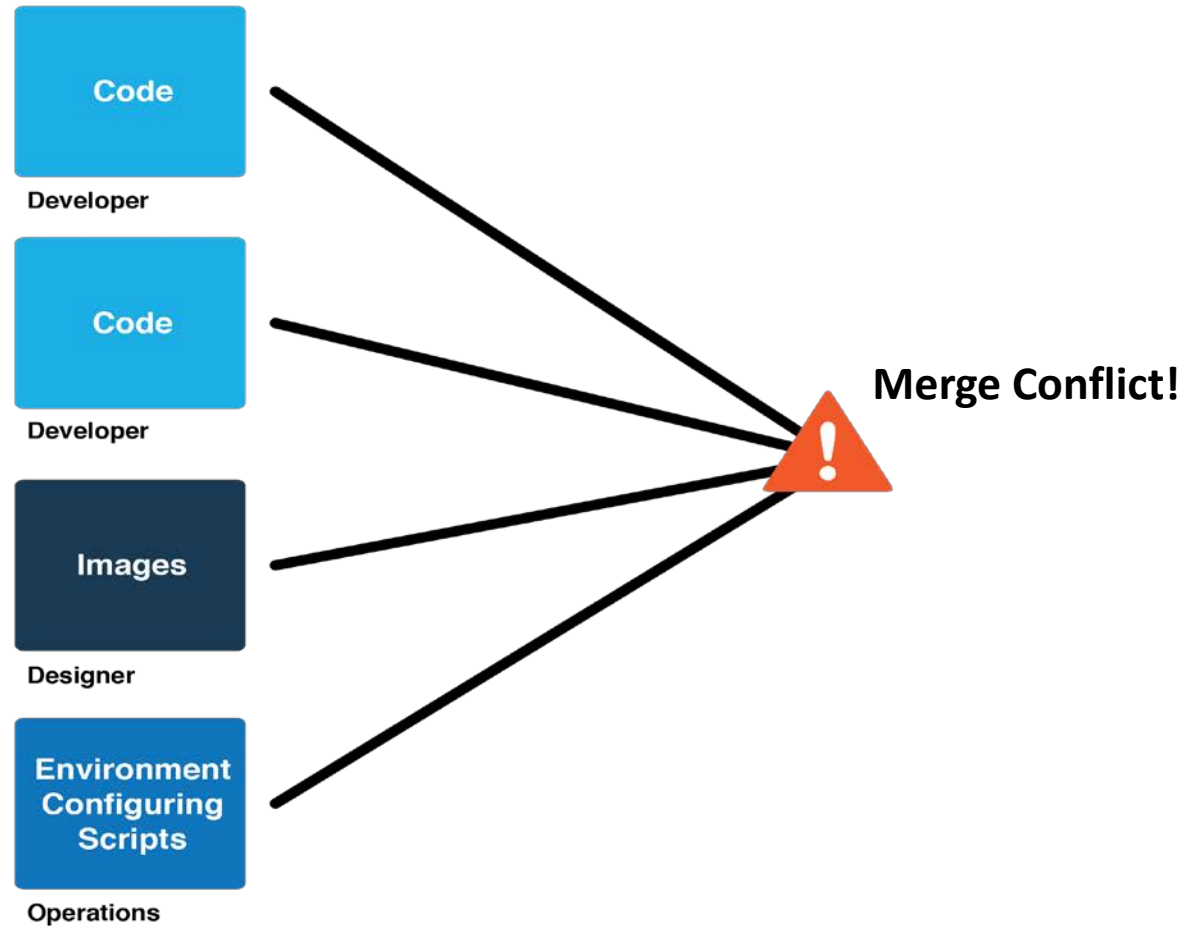
Monitoring Applications and Infrastructure

# Continuous Integration



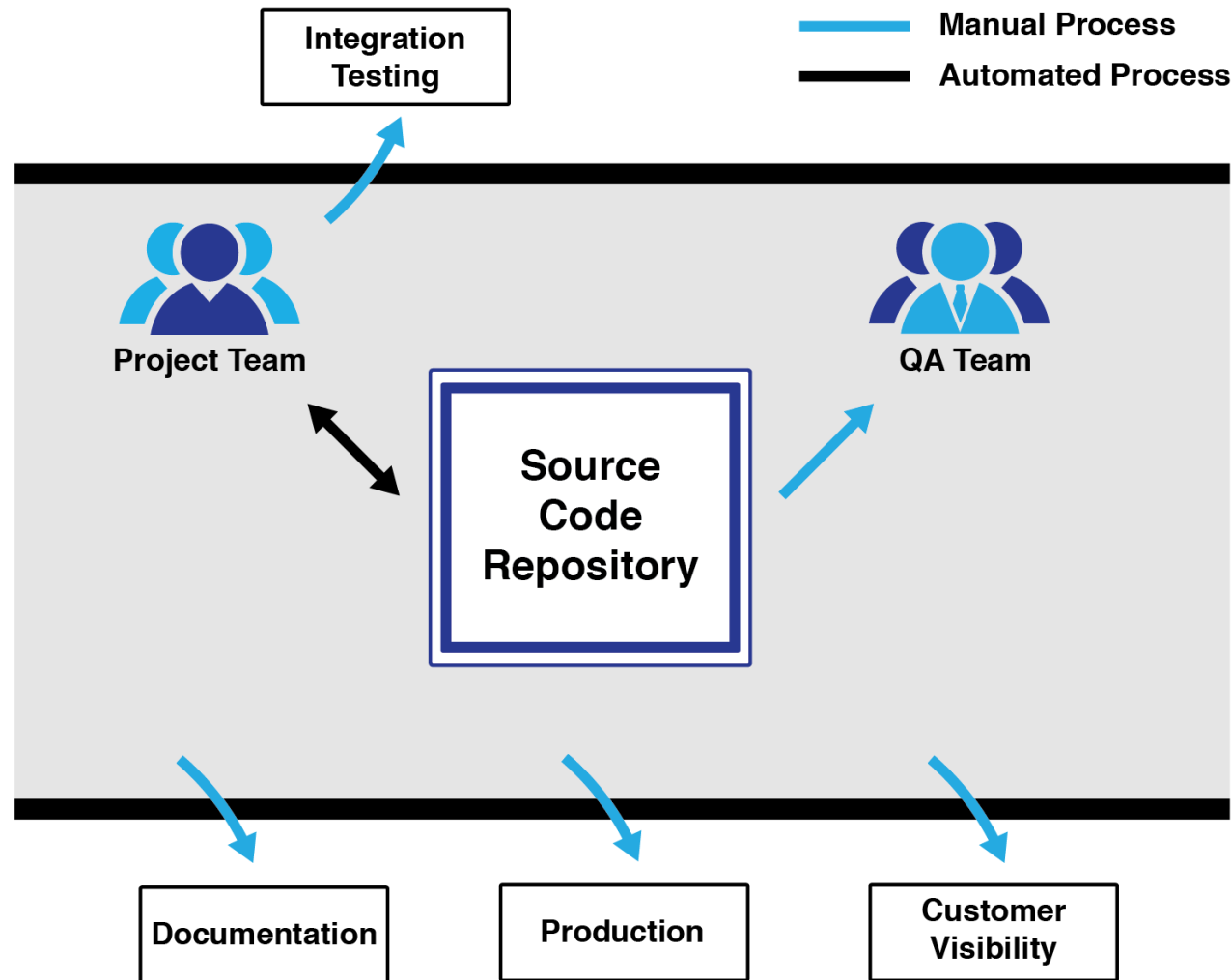
# Software projects consist of many artifacts

Integration can be challenging





# This is often a manual process



# Manual Integration is Flawed

Human-driven processes are...

- Infrequent
- Expensive
- Repetitive
- Error-prone

This leads to:

**Disjointed** activities / components

**Slow**, unreliable, costly reporting and failure recognition

**Lack of transparency** of problems

**Integration Hell**

## Polling Question

Do you currently implement Continuous Integration in your development cycle ?

1. Yes

2. No

# Automating Integration Fixes These Issues

## Automation...

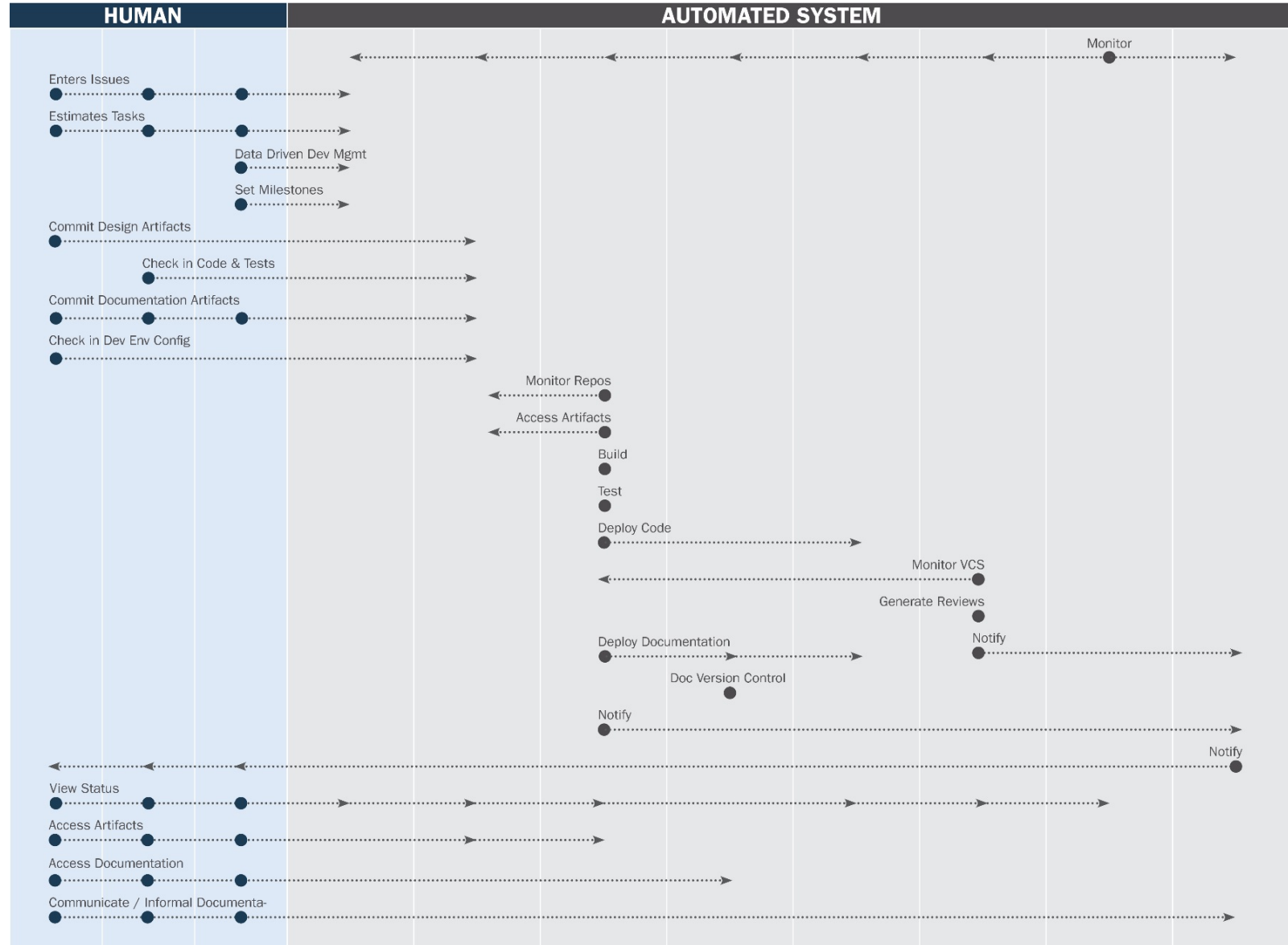
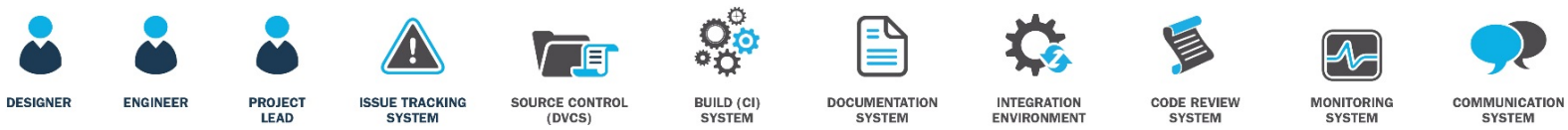
**Removes inefficiencies** due to human-driven process

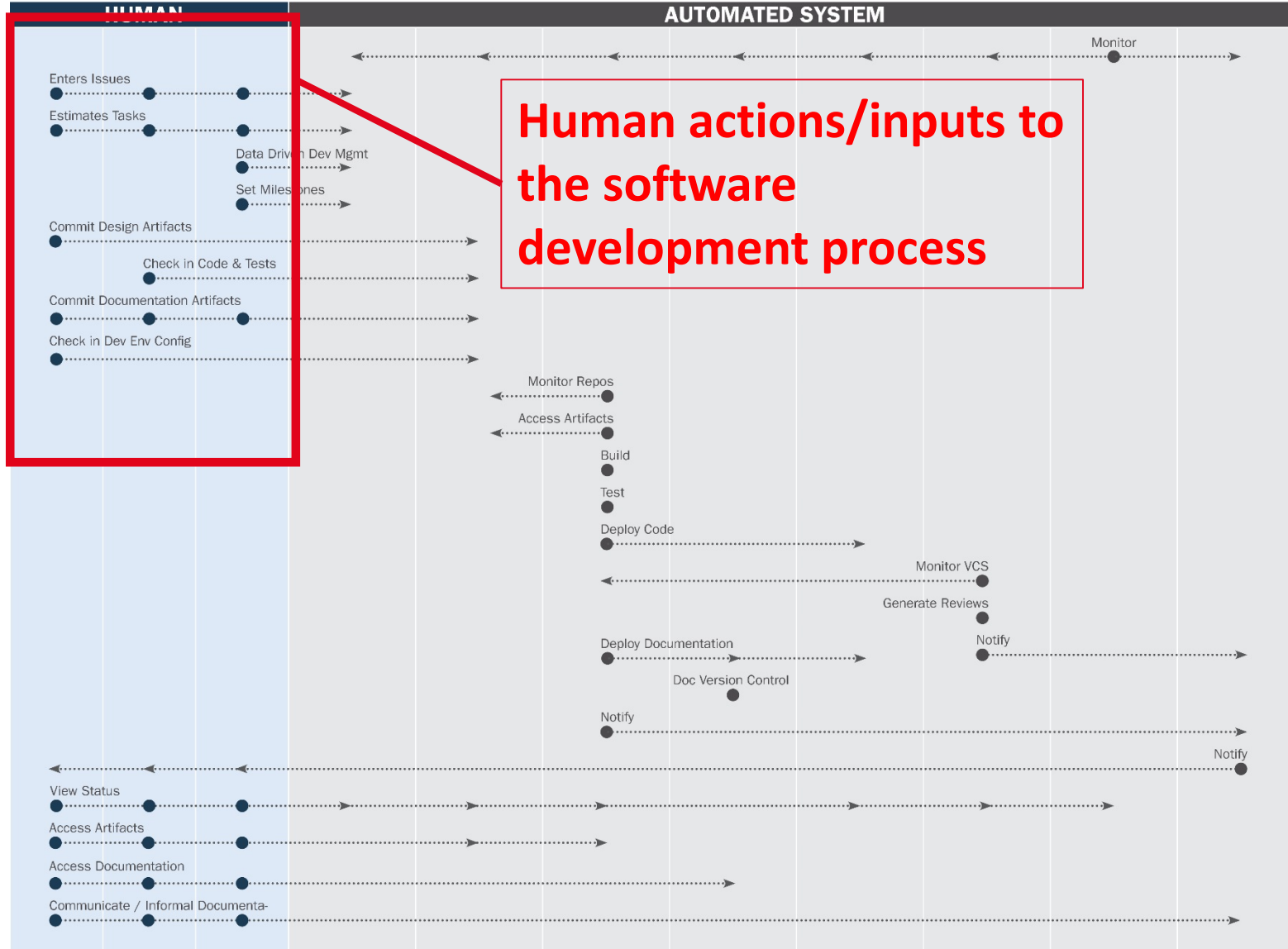
**Standardizes** artifact submission process

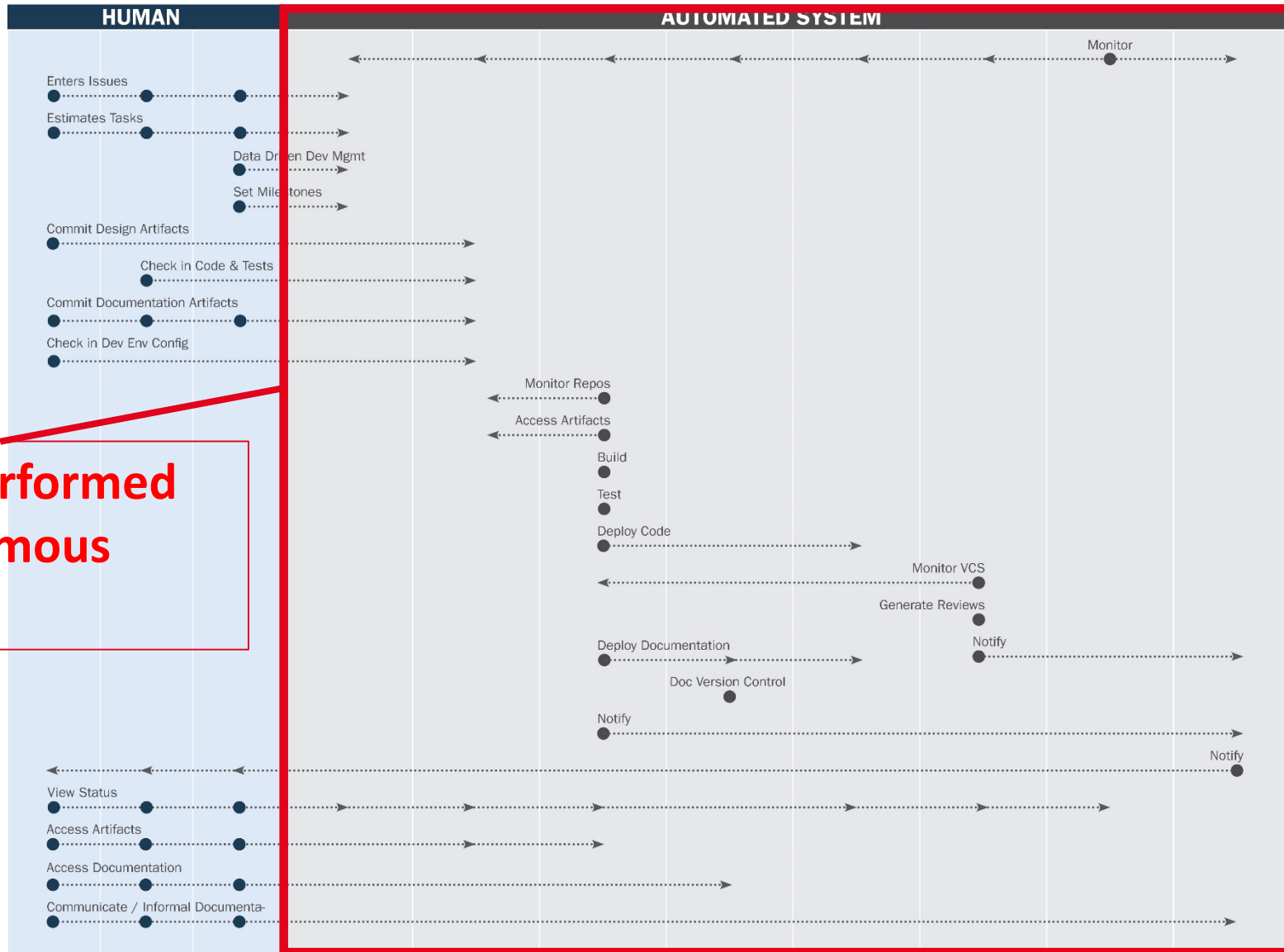
Guarantees **consistent results**

Allows team to **fail fast** (and fix fast)

**Reduces pain** of integration







**Actions performed by autonomous systems**

# *Continuous* Integration is Even Better

Continuous Integration uses a **build server** to...

Integrate artifacts on **every change**

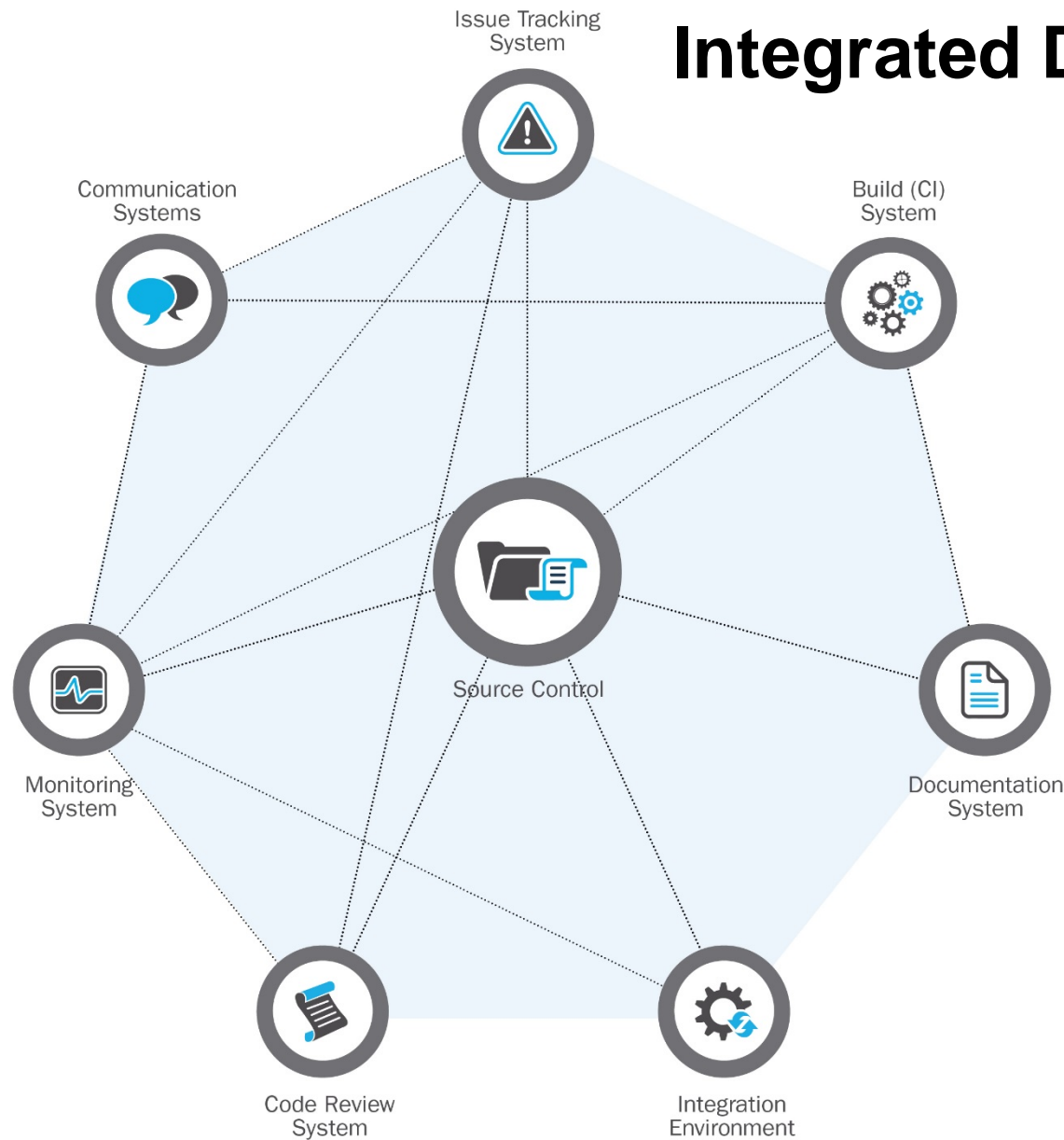
Give team with **immediate notification** of failure or success

**Require issues be fixed** before moving forward

**Enforce standards** (can fail based on quality as well as functionality)

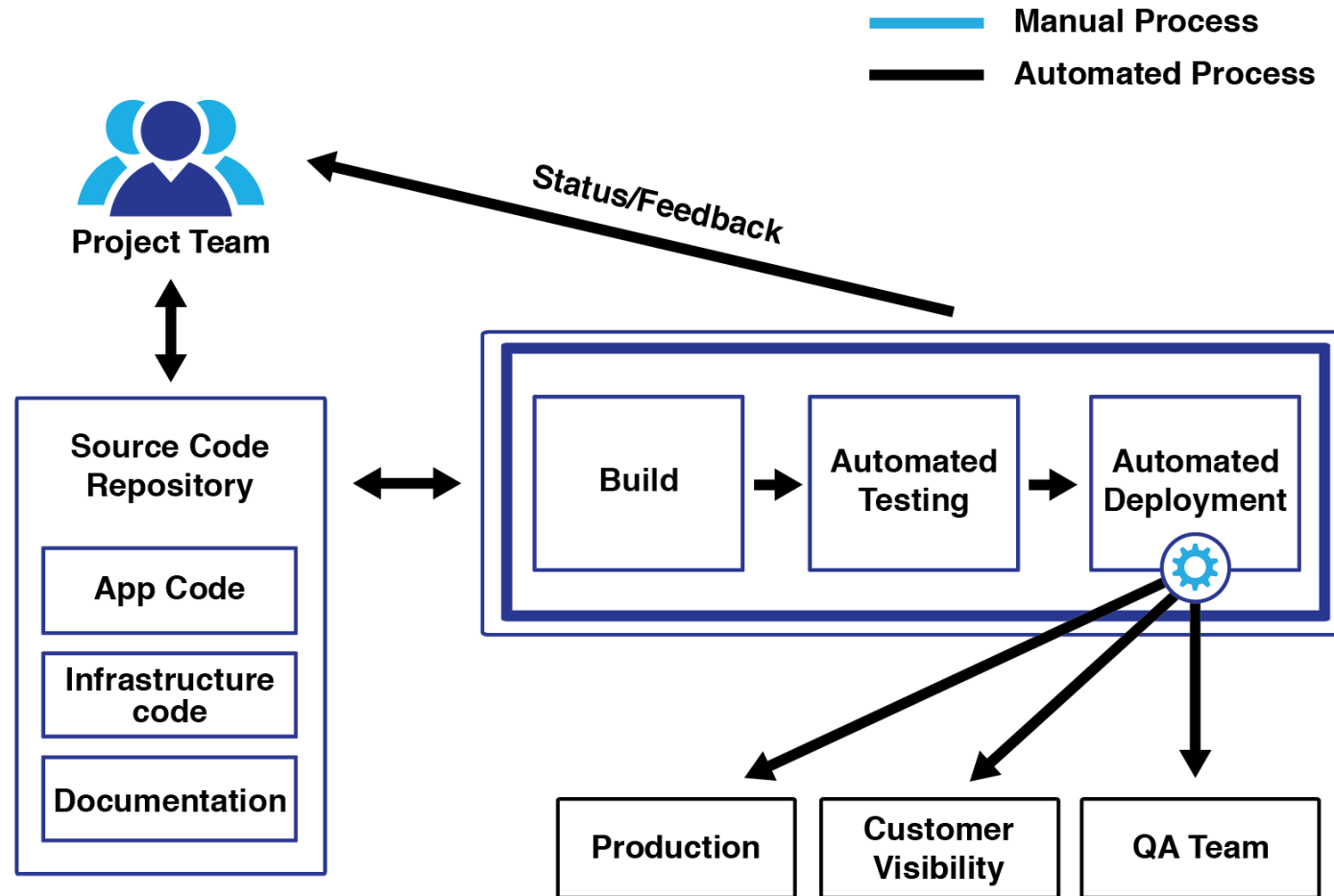


# Integrated Development pipeline



**Integration and communication**, even among tools, is the key!

# Continuous Integration (CI) Model



# Fail the Build When Software is Not Good Enough

Don't just configure failure for compile/build errors!

Want 90% test coverage? **Fail the build if code base is <90% covered**

Want all DB queries < 2sec? **Test them, and fail the build otherwise**

Want to make sure code conforms to style guide? **You guessed it...**

**CI is your best tool to enforce quality standards**

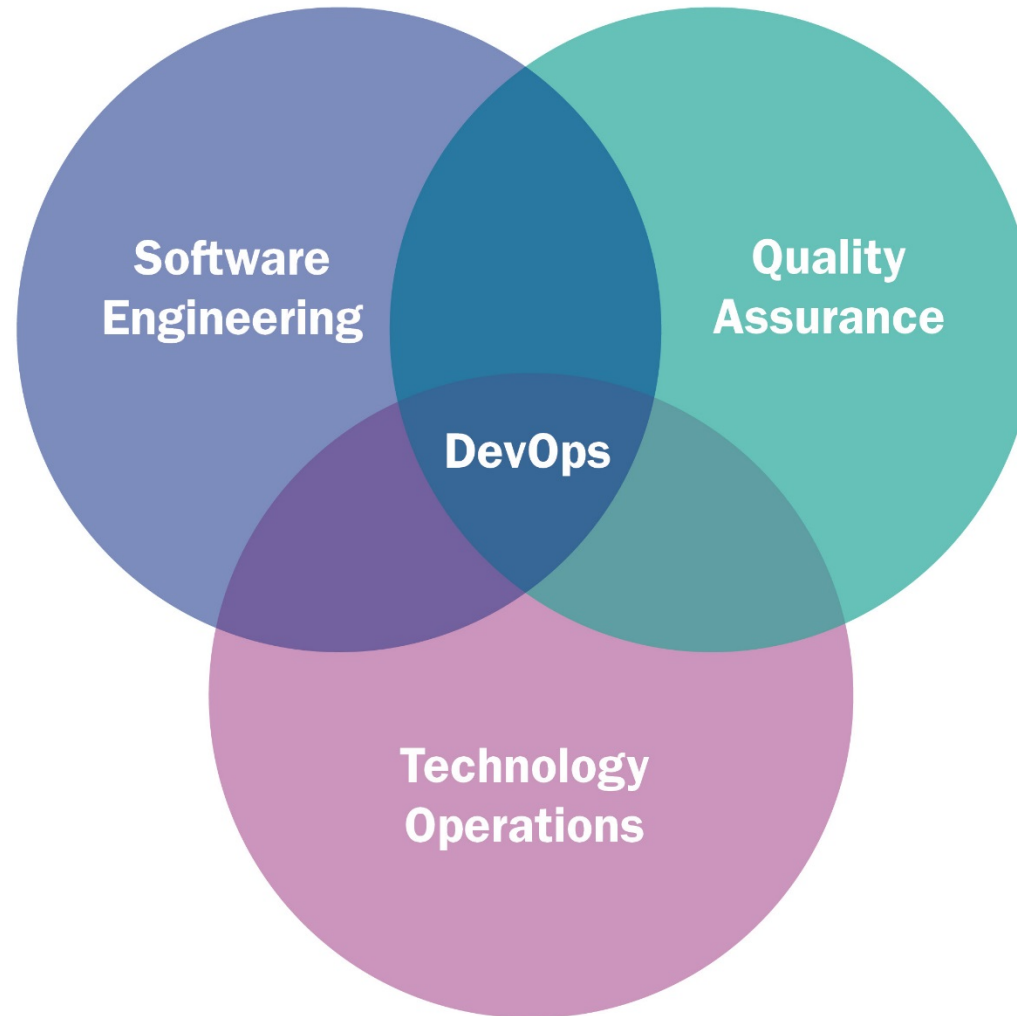
Secure DevOps  
**Integrating Security practices into DevOps**



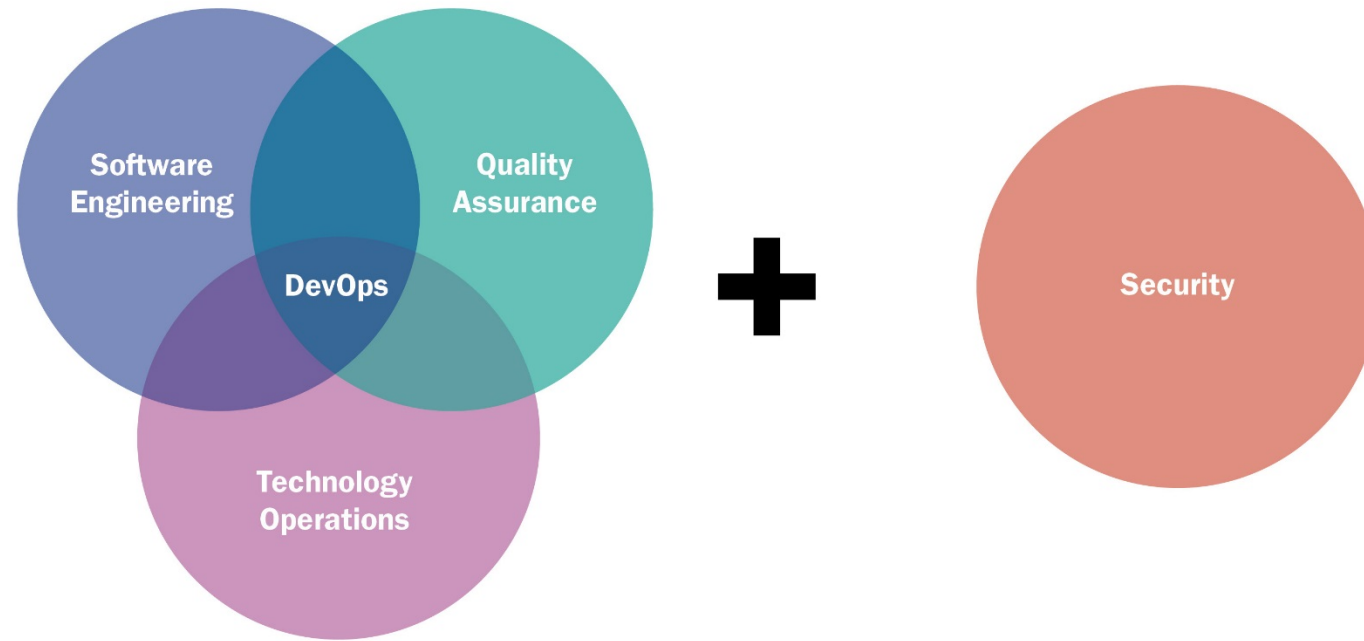
# Team Composition

<b>Developers</b>	<b>IT Ops</b>	<b>QA</b>	<b>Security Team</b>
Features Quality Attributes Efficiency Performance Users Authentication Authorization	<ul style="list-style-type: none"><li>• Deployment</li><li>• Maintenance</li><li>• Updates</li><li>• Change policy</li><li>• Failure</li><li>• Data loss</li><li>• Risk prevention</li></ul>	<ul style="list-style-type: none"><li>• Testable</li><li>• Issue tracking</li><li>• Bug Reports</li><li>• Usability</li><li>• Help Desk</li></ul>	<ul style="list-style-type: none"><li>• Data Privacy</li><li>• Intrusion detection</li><li>• Threat vectors</li><li>• CVEs</li><li>• Package security</li><li>• Authentication</li><li>• Authorization</li><li>• Security Standards Compliance</li></ul>

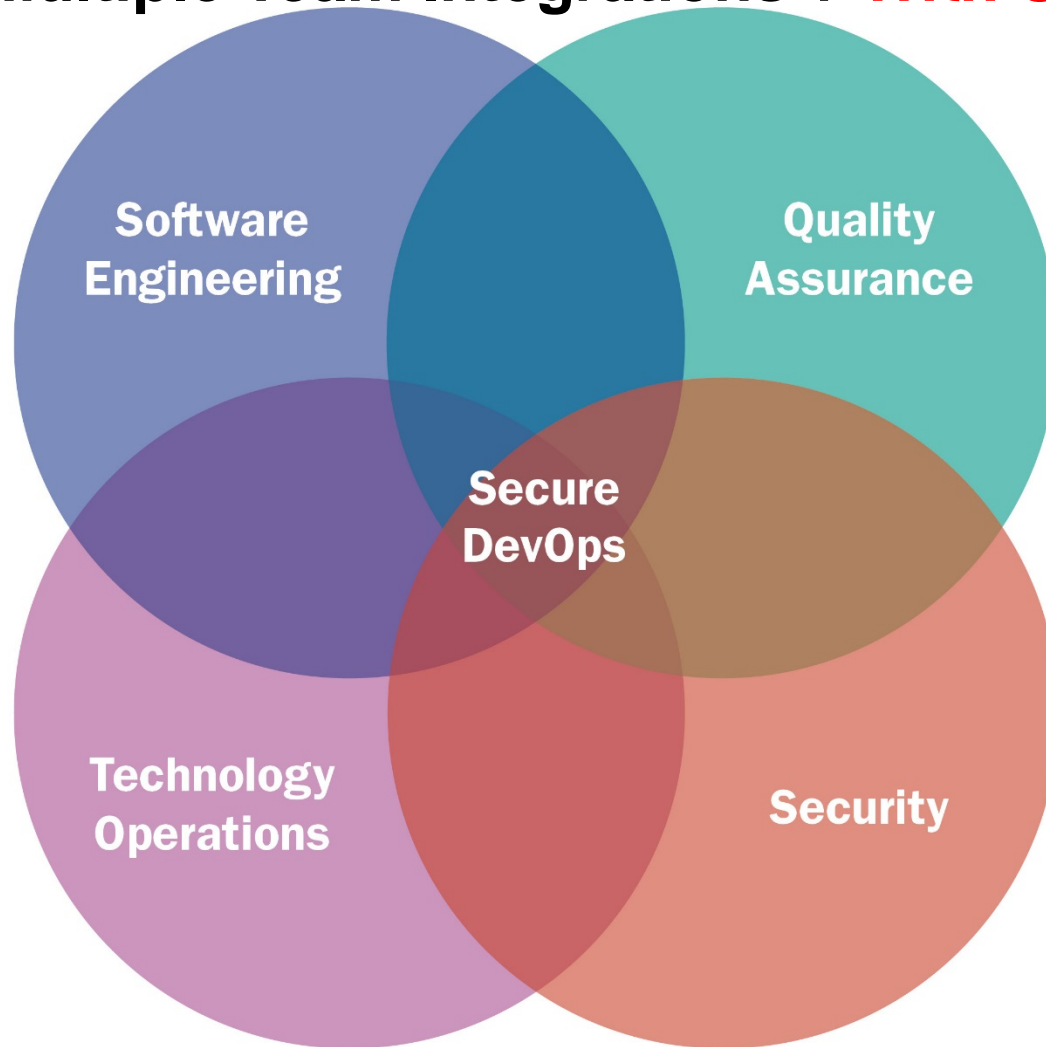
# DevOps: Multiple Team Integrations



# DevOps: Multiple Team Integrations + *With Security Team*



# DevOps: Multiple Team Integrations + *With Security Team*



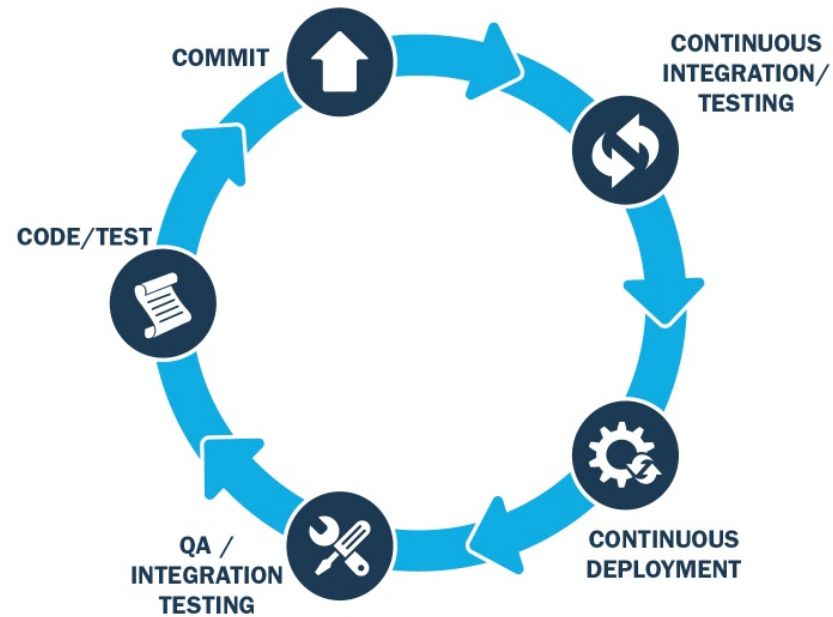


## Polling Question

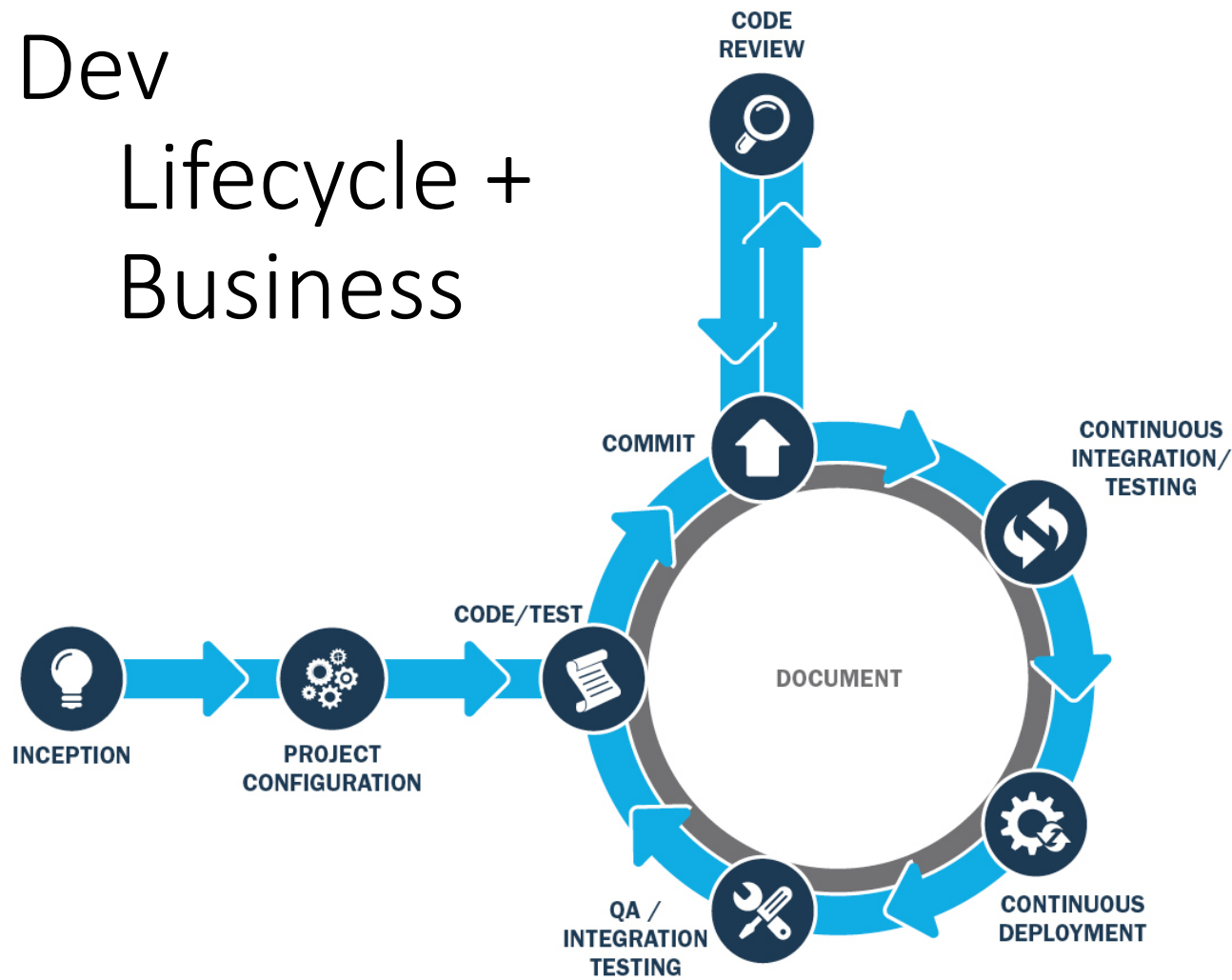
At what point do you consider security?

- a. At the very beginning
- b. Sometimes in the middle
- c. Toward the end
- d. Not at all

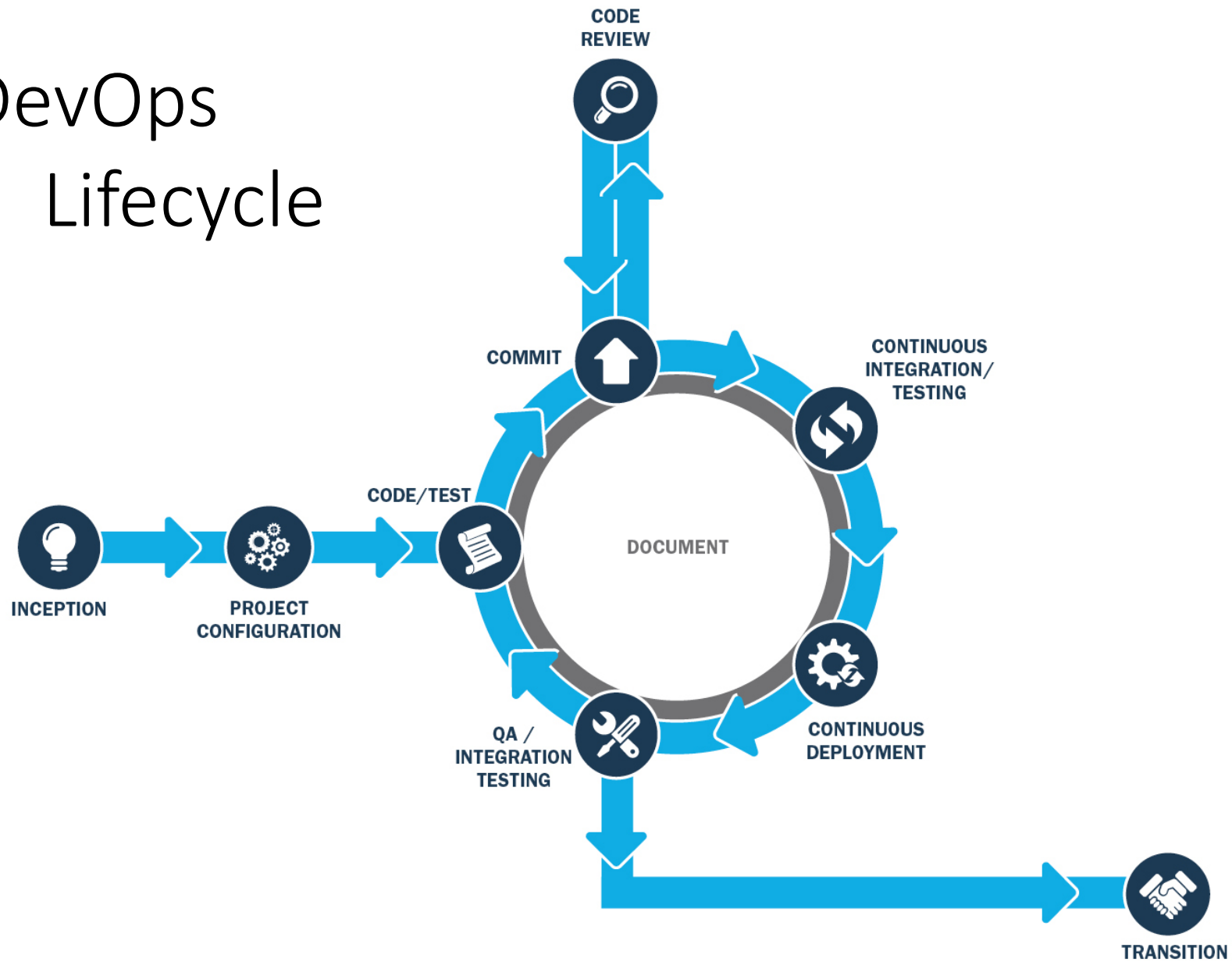
# Dev Lifecycle



# Dev Lifecycle + Business

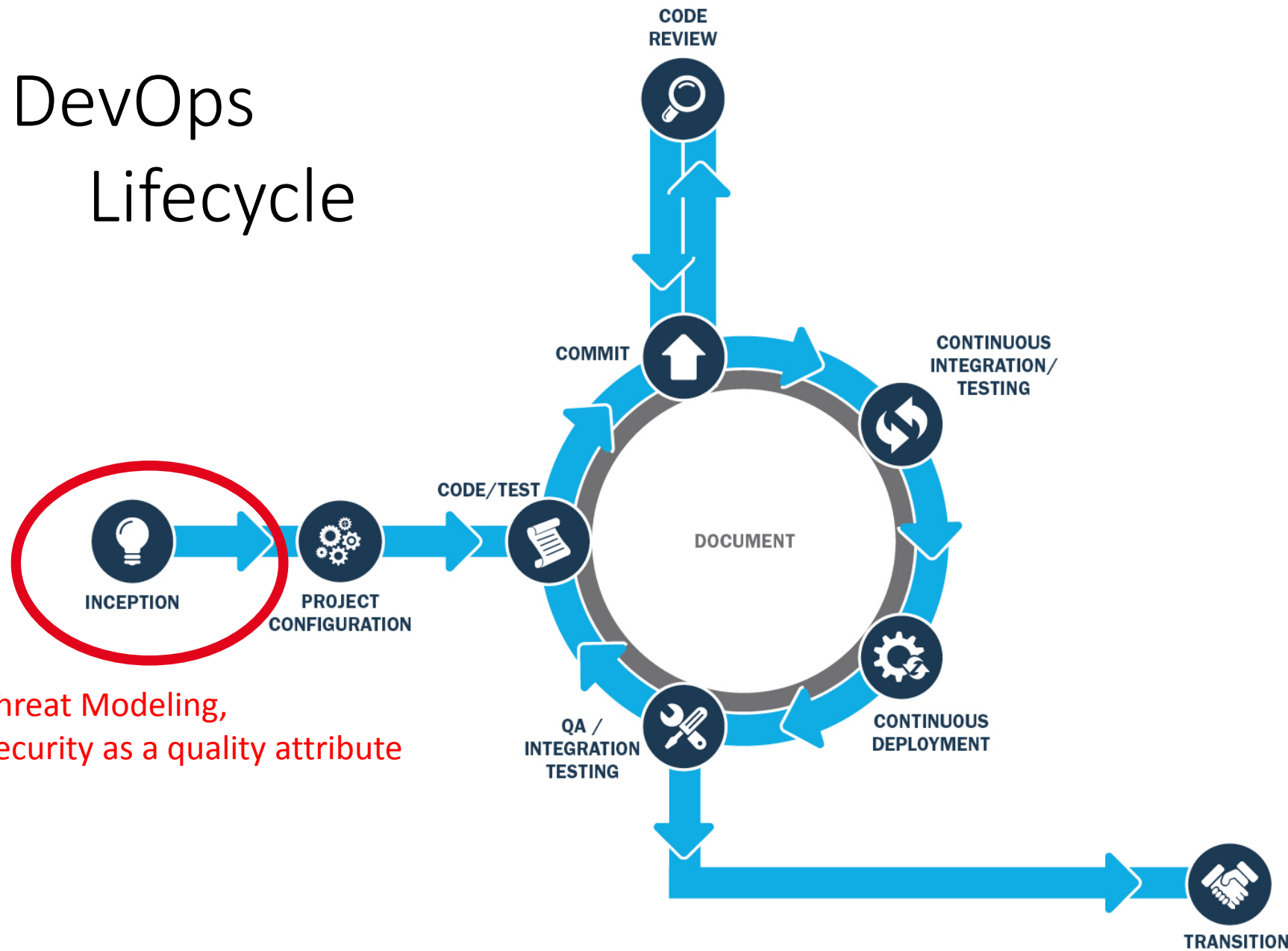


# DevOps Lifecycle



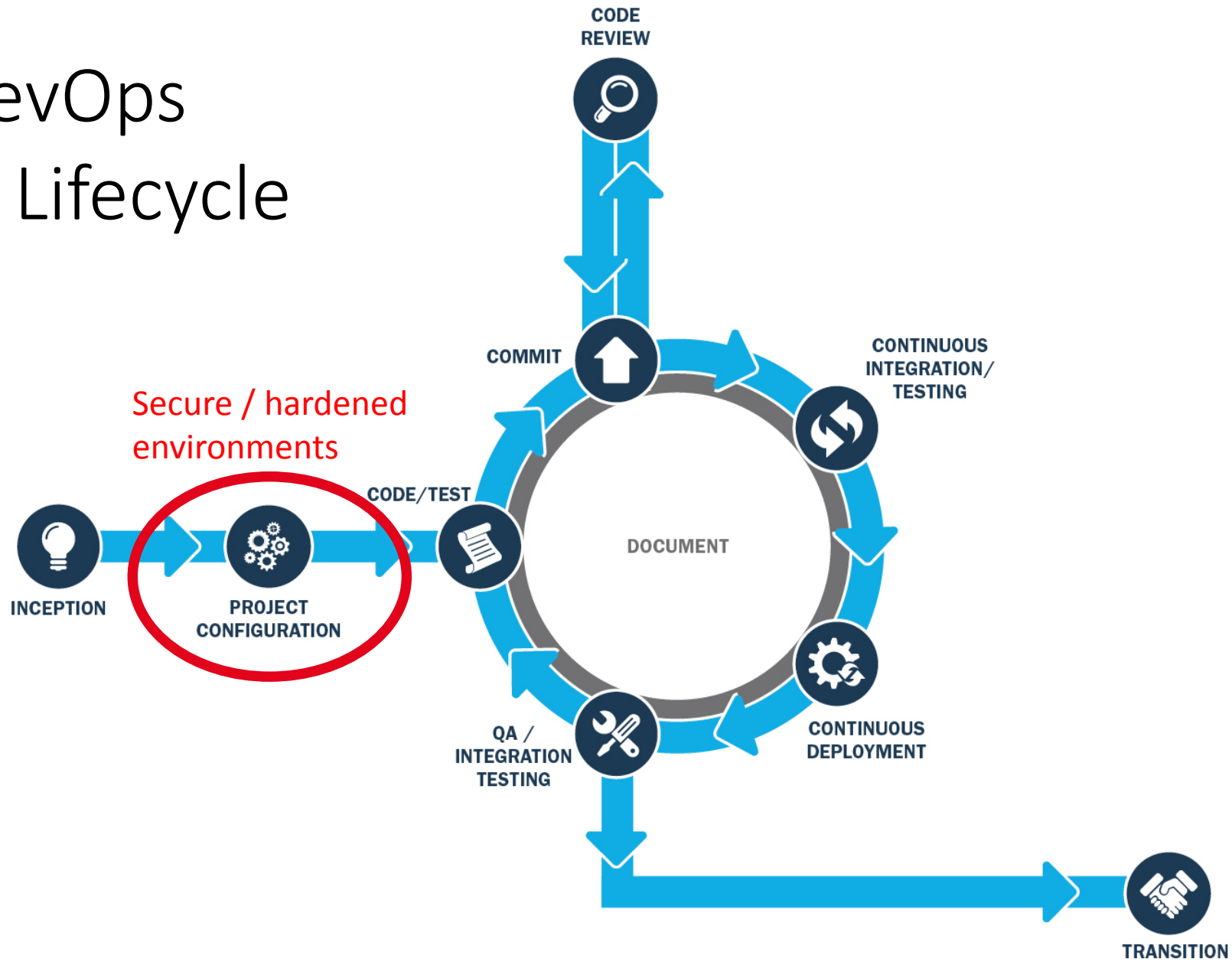
Where are opportunities for *security processes*?

# DevOps Lifecycle

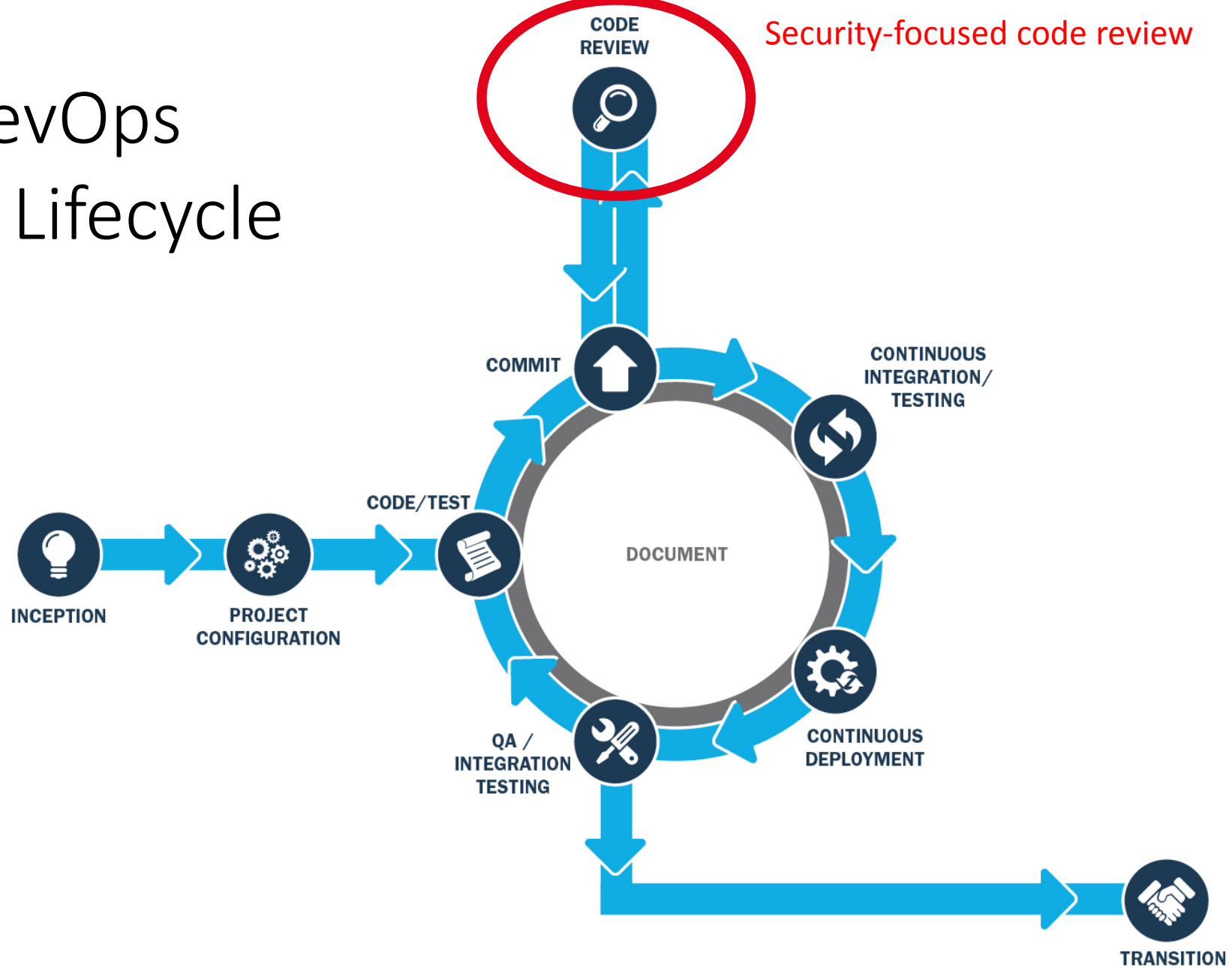


Threat Modeling,  
Security as a quality attribute

# DevOps Lifecycle

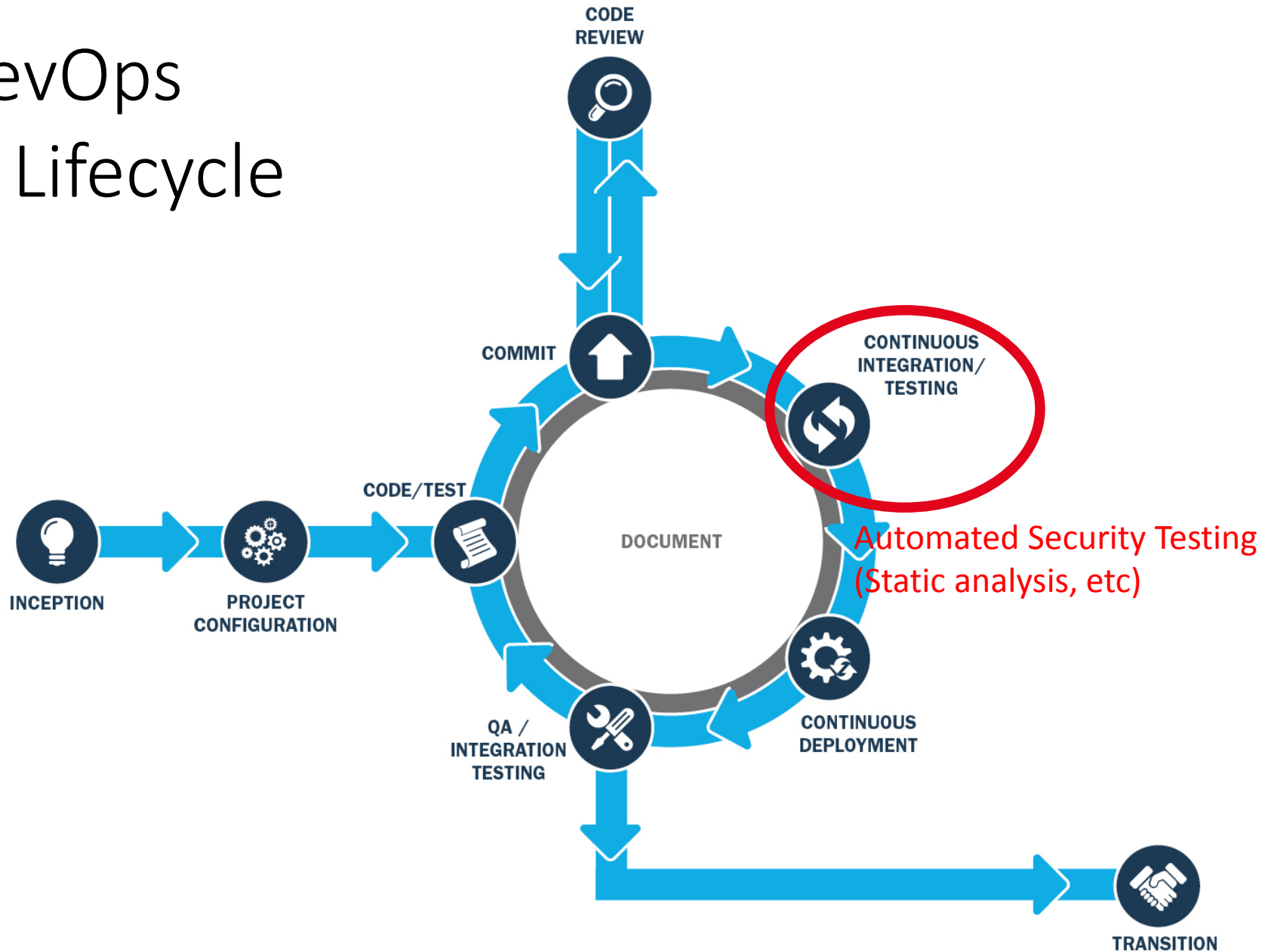


# DevOps Lifecycle

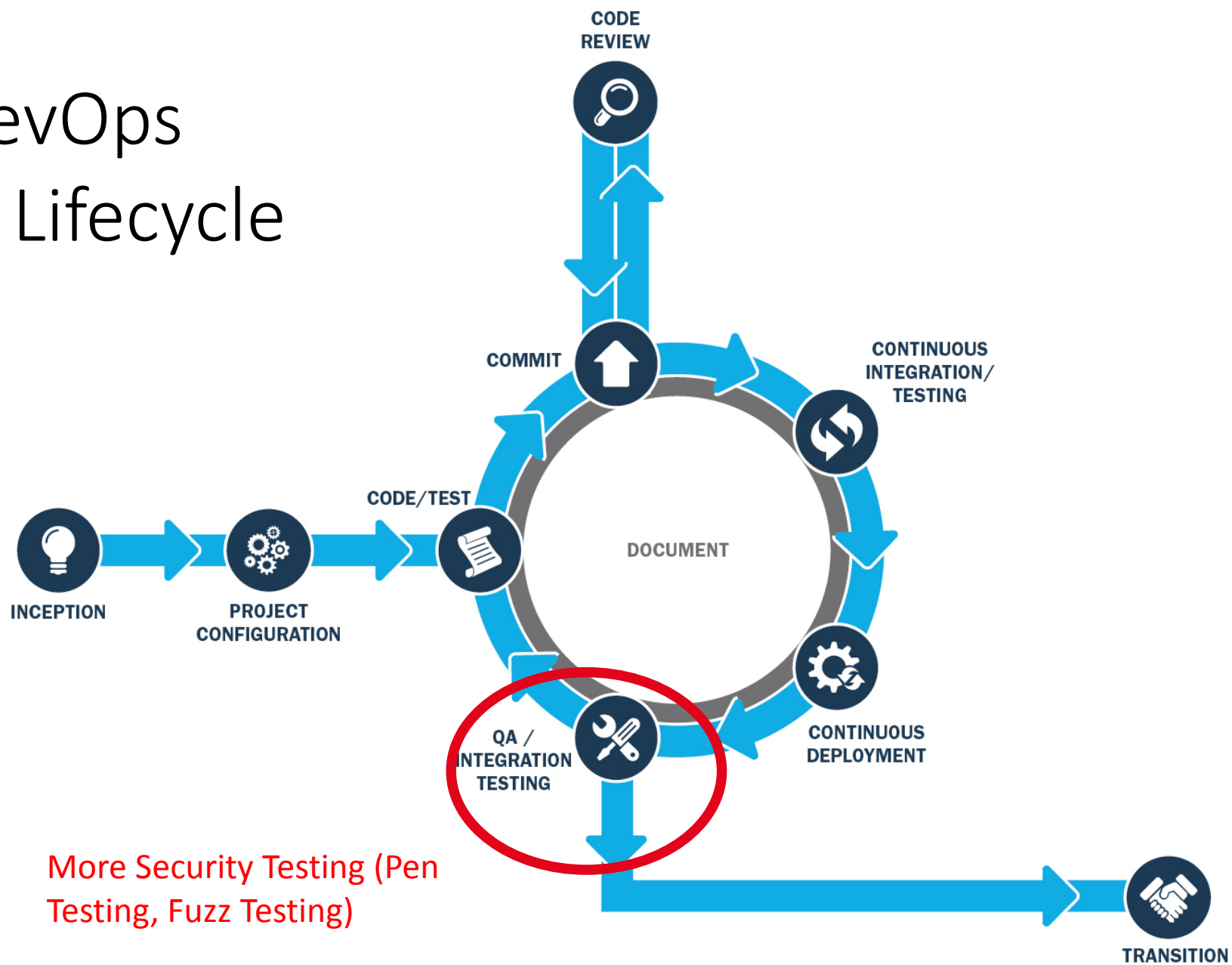




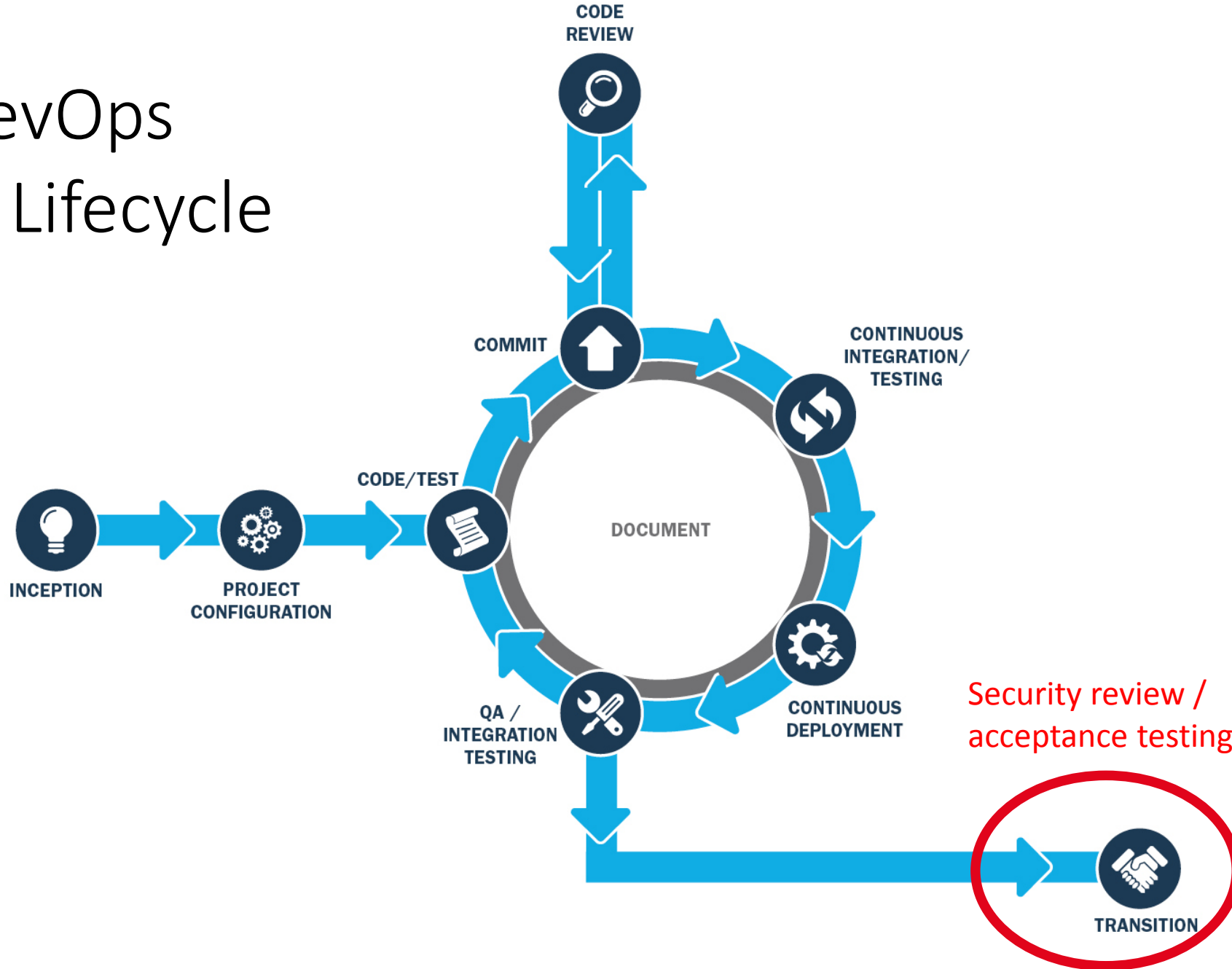
# DevOps Lifecycle



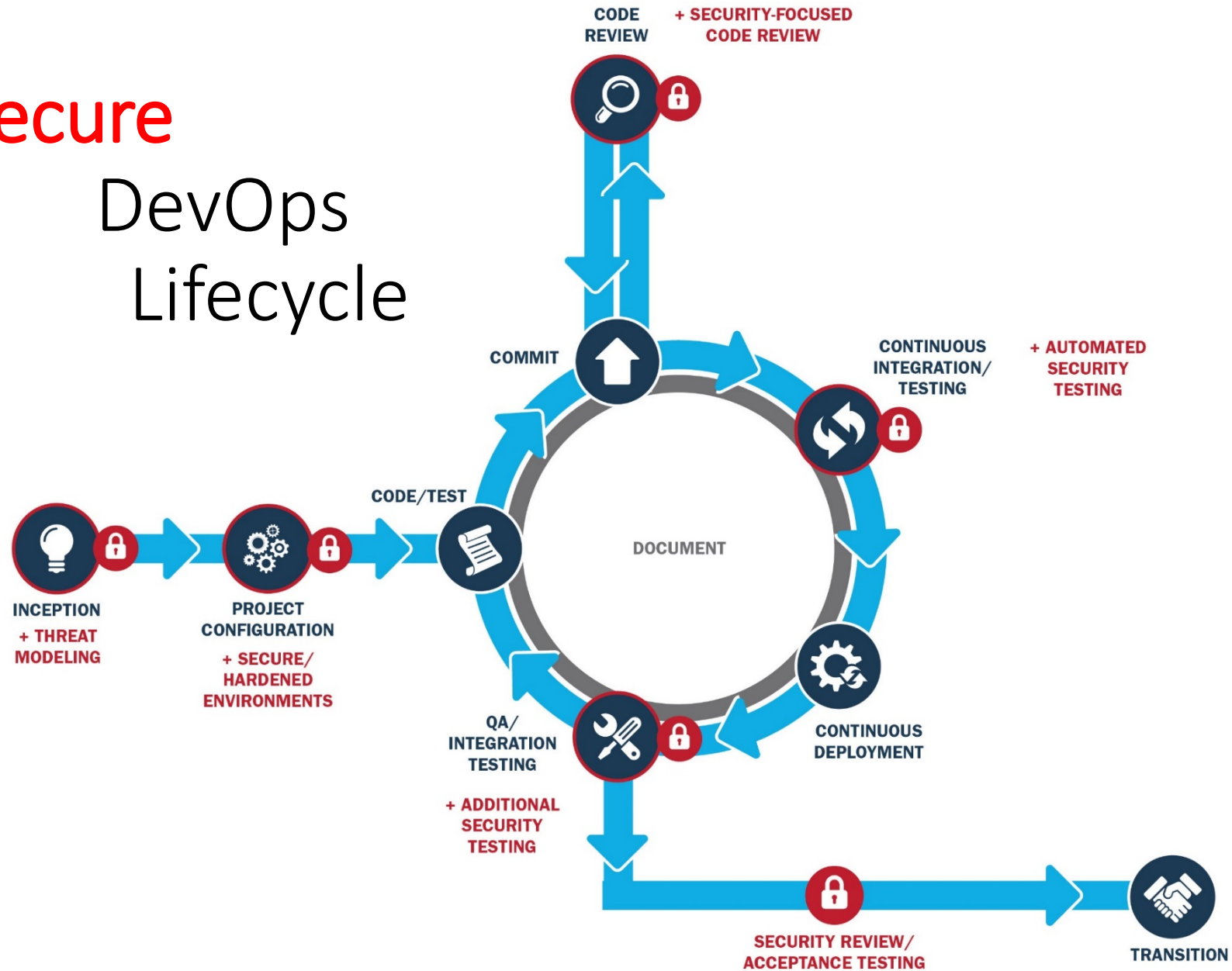
# DevOps Lifecycle



# DevOps Lifecycle



# Secure DevOps Lifecycle



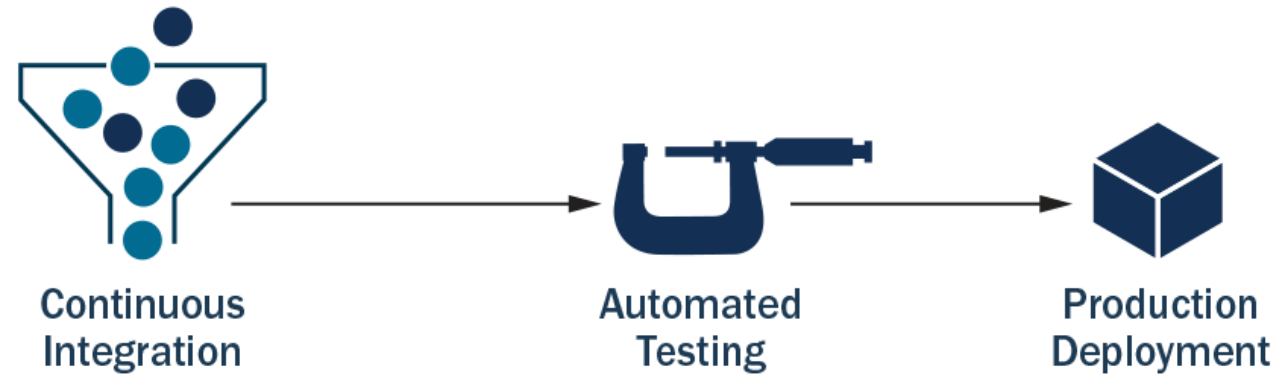
**Security must be addressed without breaking  
the *rapid delivery, continuous feedback* model**

Secure DevOps

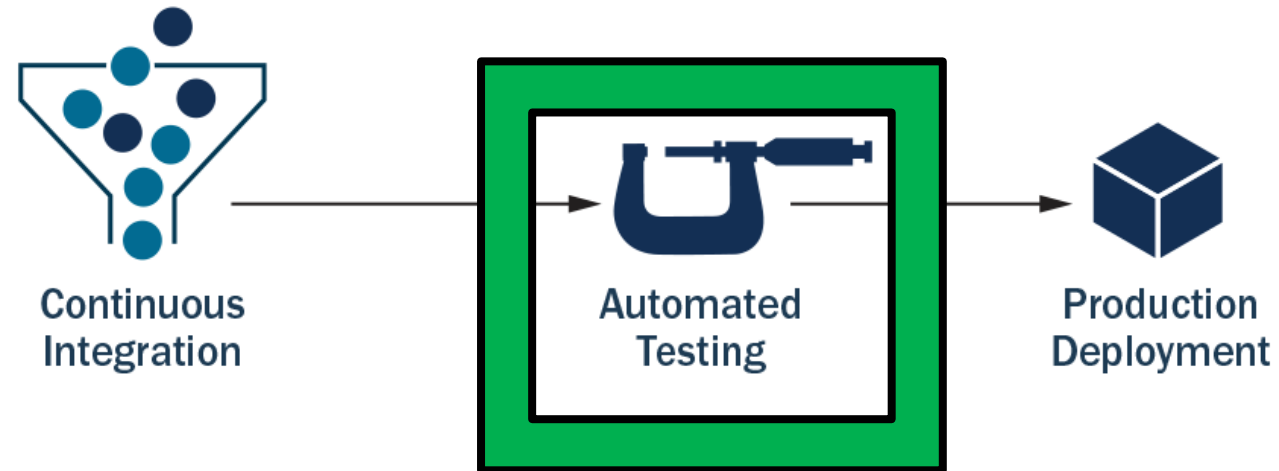
# A typical scenario



# Static Code Analysis



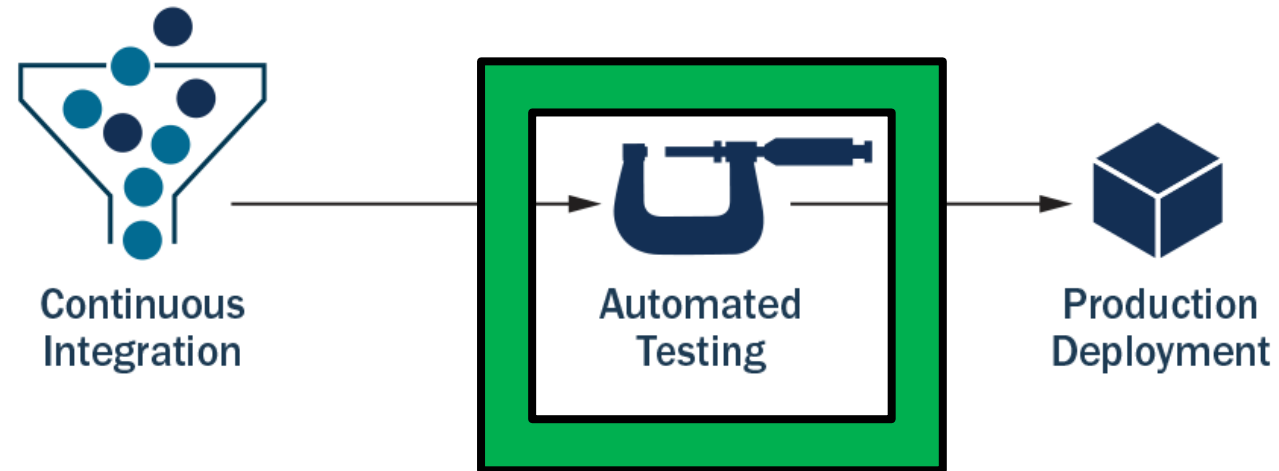
# Static Code Analysis



Tools Vary by Technology

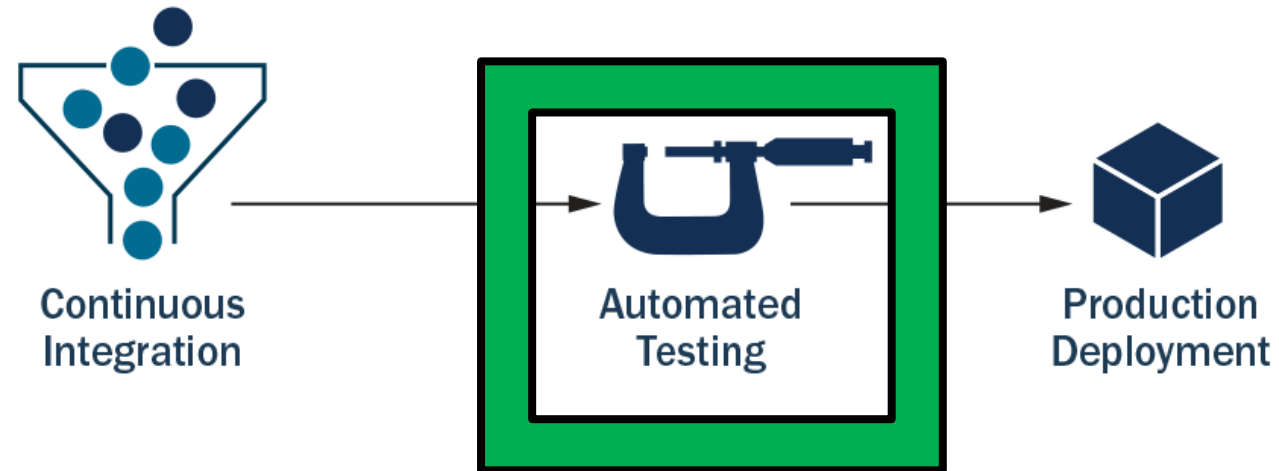


# Static Code Analysis



Regular Expressions

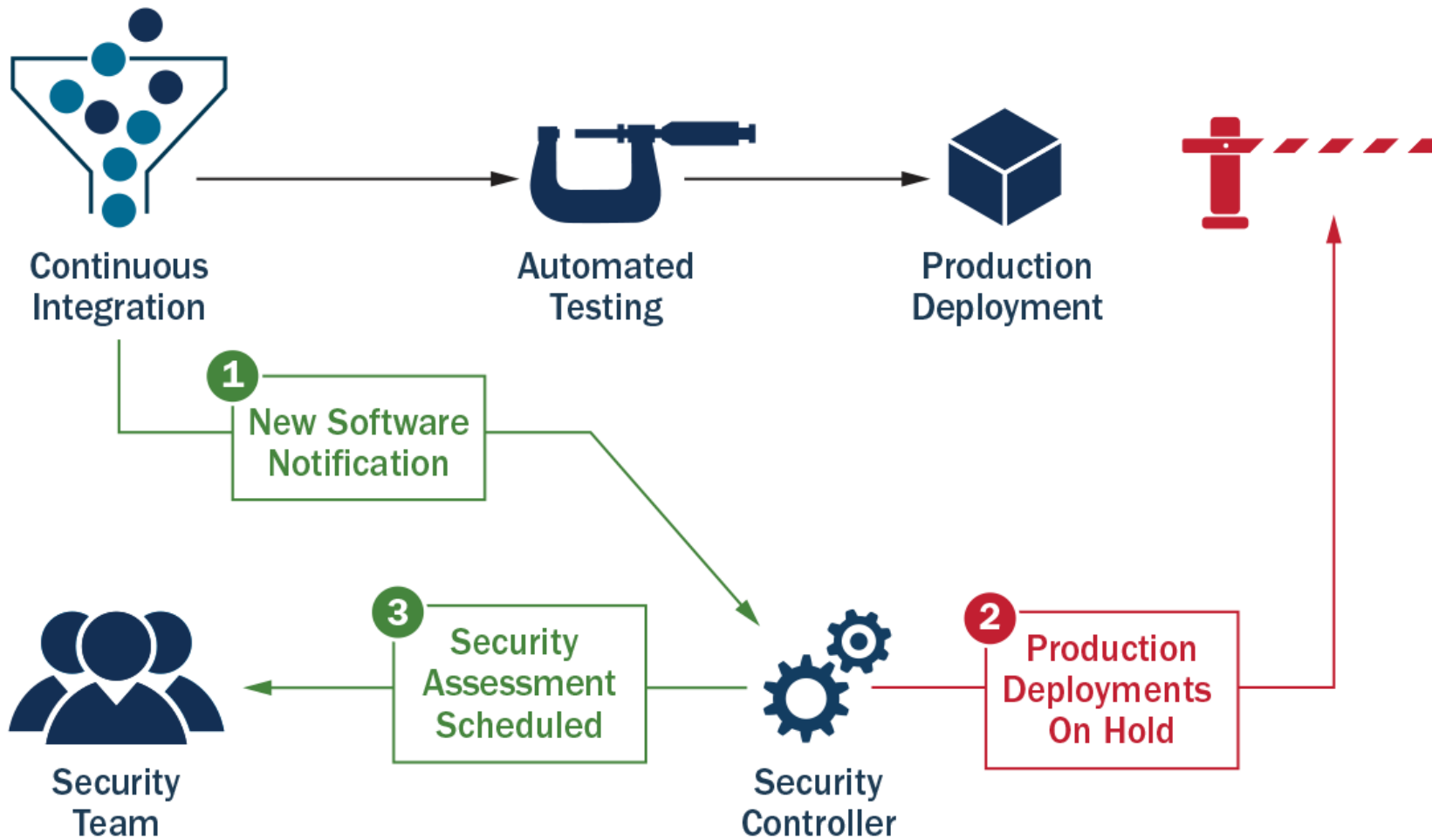
# Static Code Analysis



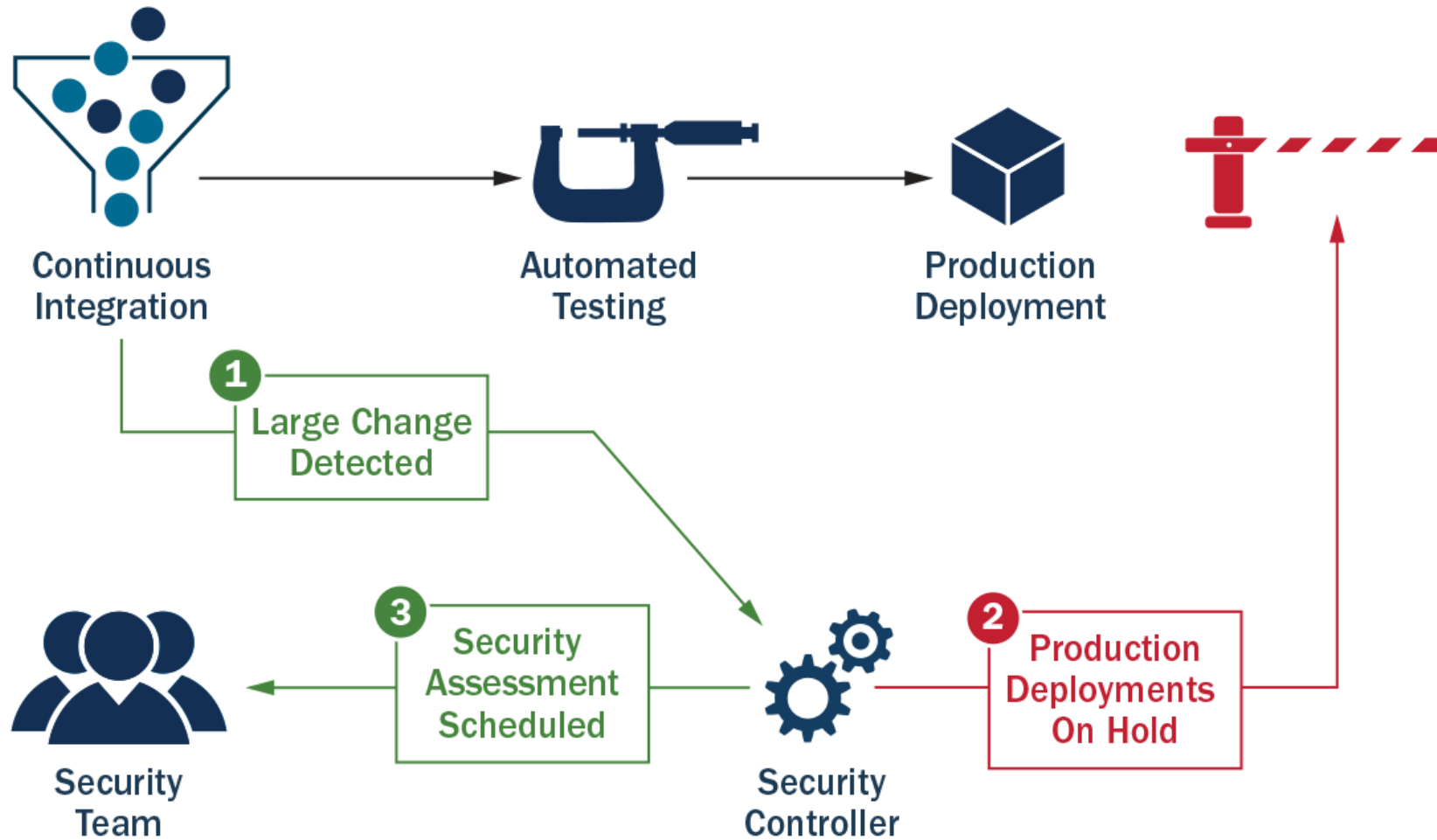
Wrappers

# Manual Security Assessments

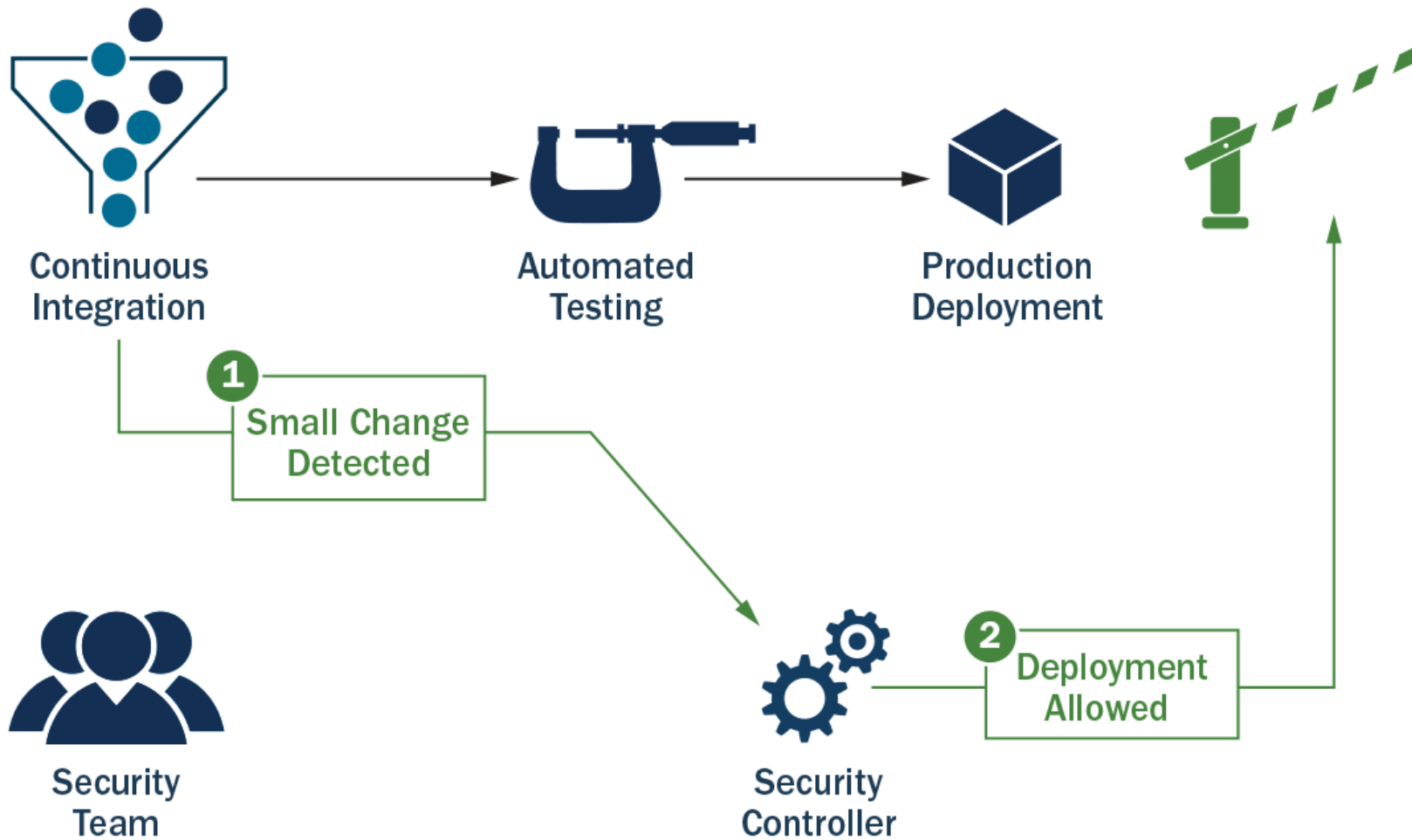
- Centralize information needed to conduct assessments
- Pick and choose your battles
- Integrate your tools as much as possible
- Capture outputs from tools in a central repository













## More on SEI DevOps Blog

<https://insights.sei.cmu.edu/devops>

# Contact Information

Hasan Yasar

[hyasar@cmu.edu](mailto:hyasar@cmu.edu)

 <https://www.linkedin.com/in/hasanyasar>



**Q&A**



**Software Engineering Institute**

**Carnegie Mellon**