CYBER JOURNEY

CYBER JOURNEY

OWASP 2025
ITALY DAY

19th JUNE 2025
START H 16.00

FRONTEMARE CAGLIARI
SARDINIA - ITALY

2025, June 19th

DevSecOps...Where should we shift?

Enrico Trasatti – Francesco Favara

Security Services & Solutions, Sogei

Speaker

Enrico Trasatti

Who is Sogei?

As the Ministry of Economy and Finance's exclusive technology partner, it builds systems, applications, and services that fully automate and digitize the operational and management processes of the Ministry, the Court of Auditors, tax agencies, and other public administrations.

Main Partner

Agenzia Entrate · ADM · Agenzia del Demanio · Riscossione Agenzia Entrate · Corte dei Conti · MEF · Ministero dell'Interno · Ministero dell'Ambiente e della Tutela del Territorio e del Mare · MUR · Ministero della Giustizia · Ministero della Salute · Presidenza del Consiglio dei Ministri · ACN · MIM Ministero dell'Istruzione e del Merito · Equitalia Giustizia · Ministero della Cultura · consip

What's my role?

sogei

Cybersecurity

Security Services & Solutions

- Red Teaming & Penetration Test
- Vulnerability Assessment & Management
- Cloud & Container Security
- DevSecOps

A complex operational scenario

sogei

20+
Application Domain

10.000+
Annual releases

1500+
Developer

5000+
Repository

Different Programming Languages:

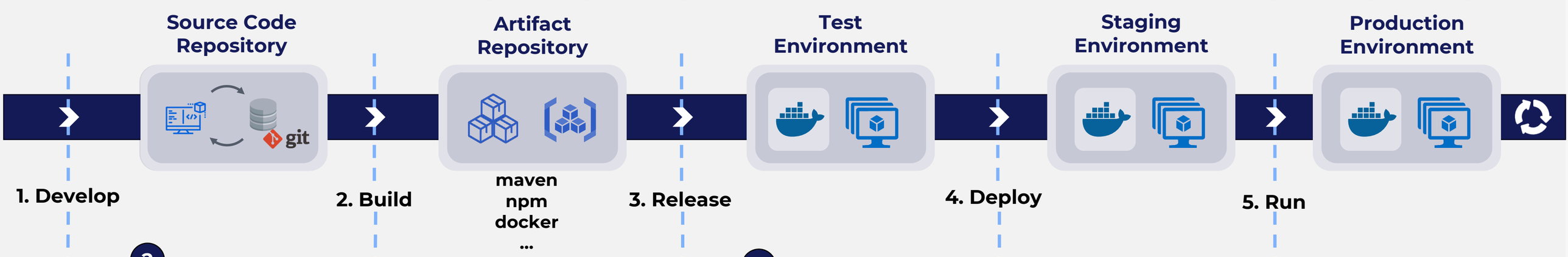
Java
Javascript
.NET
...

500+
Web Services

Deploy:
Legacy & Microservices

40M+
Line of Code

The Theory: generic SDLC process



1

The **source code** is **created** by development teams and the output is stored in the **Source Code Repository**, which manages and **tracks all versions** and **modifications**.

2

Software compilation and **packaging** phase. The generated **artifacts** are **deposited** in the **Artifact Repository**, ready for distribution.

3

The **application** is **released** and **distributed** in the **Test Environment**. In this environment, **software quality** is **verified** through appropriate **functional** and **non-functional tests**.

4

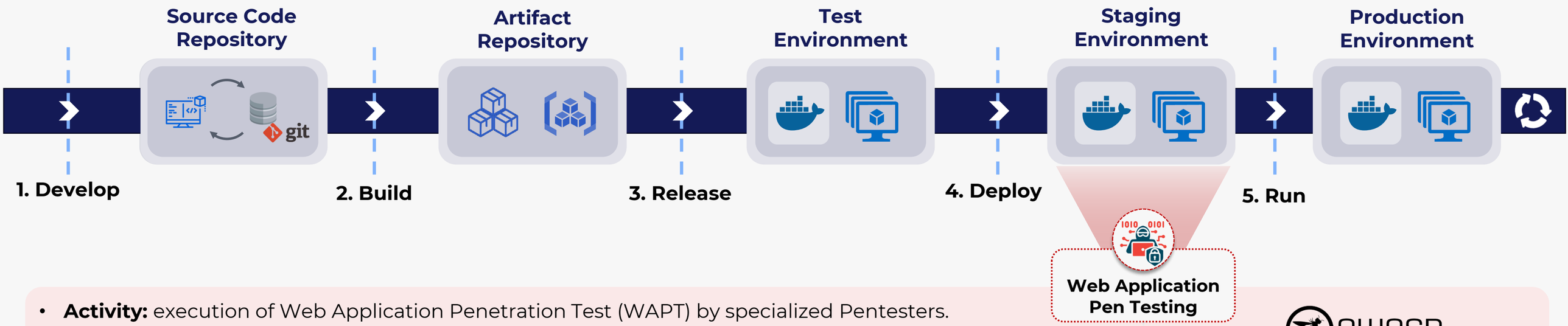
Deployment phase in the **staging environment** to perform **final testing before** moving to **production**.

5

Operational phase where the **application** goes into **production** and the **service** is **delivered** to end users.

DevOps

The Theory: WAPT execution in SDLC process



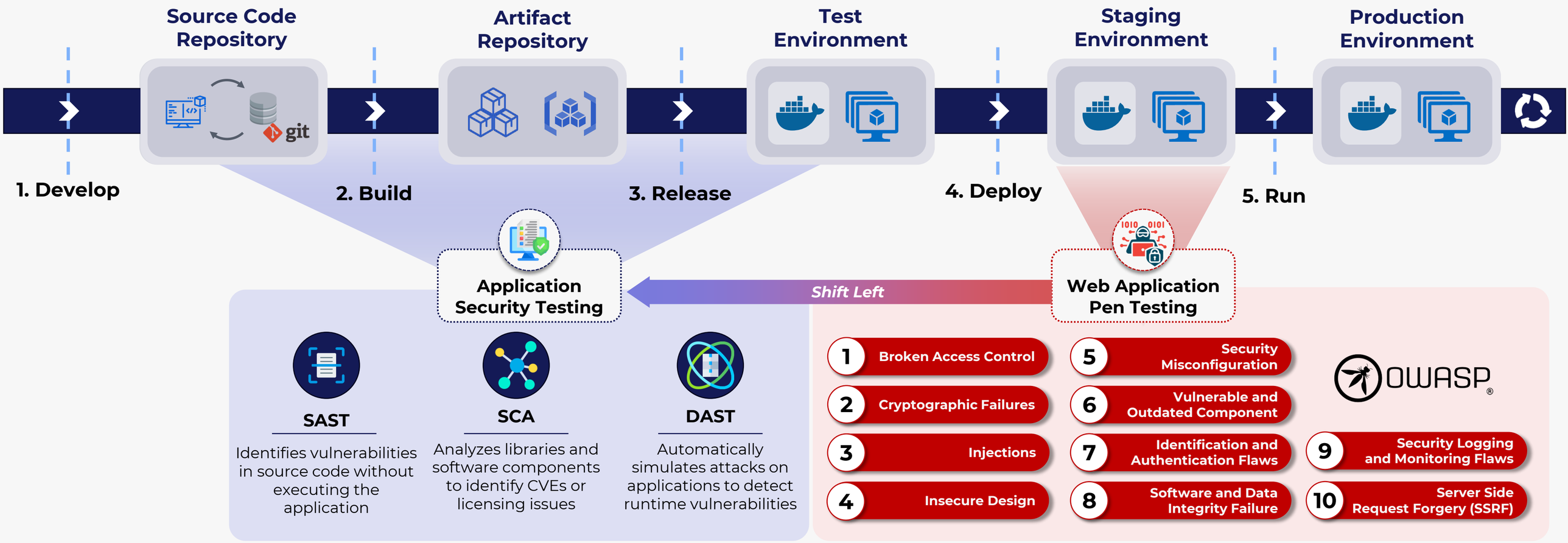
- **Activity:** execution of Web Application Penetration Test (WAPT) by specialized Pentesters.
- **Reference classification:** OWASP TOP 10 2021

OWASP®

1 Broken Access Control	3 Injections	5 Security Misconfiguration	7 Identification and Authentication Flaws	9 Security Logging and Monitoring Flaws
2 Cryptographic Failures	4 Insecure Design	6 Vulnerable and Outdated Component	8 Software and Data Integrity Failure	10 Server Side Request Forgery (SSRF)

DevOps

The Theory: integration of automatic AST controls in the SDLC process



DevSecOps

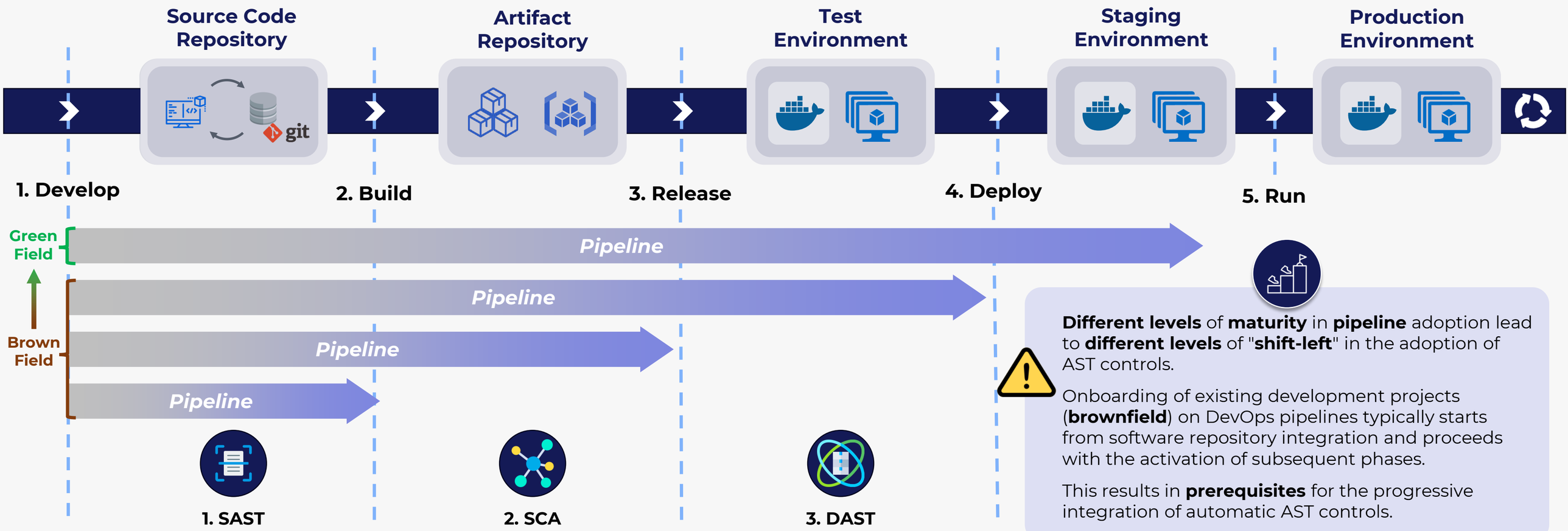
The Practice: ... some issues to address and resolve

Key Questions:

1. From a **progressive onboarding** perspective, which **controls** should you **start** with and **why**?
2. By **applying** automatic controls of **SAST**, **SCA** and **DAST/IAST** type do we have the **same** type of **coverage** as manual **WAPT**, reserving them only for particular situations?
3. If there were **vulnerabilities** by their nature **not** easily **detectable** by **automatic controls**, how to **address** and **prevent** them in the SDLC?
4. **How** to effectively **involve development teams**?
5. **How** to **ensure** that what goes into **production** has effectively **passed all controls**, while there will be many other sw components in development phase with vulnerabilities to resolve?
6. **How** to **manage** in the SDLC the **vulnerabilities** that emerge "**unattended**" **after** having put the code into **operation**? (e.g. CVEs of libraries)

Aspects to **address** with particular **attention** and **effort**

The Practice: AST controls in the SDLC ... prerequisites



Different levels of **maturity** in **pipeline** adoption lead to **different levels** of "**shift-left**" in the adoption of AST controls.

Onboarding of existing development projects (**brownfield**) on DevOps pipelines typically starts from software repository integration and proceeds with the activation of subsequent phases.

This results in **prerequisites** for the progressive integration of automatic AST controls.

DevSecOps

Aspects to **address** with particular **attention** and **effort**

The Practice: AST controls in the SDLC ... prerequisites

	SAST	SCA	DAST
Pro	▶ **Greater coverage** of projects ▶ **Useful indications** to the developer on the code	▶ **Ease of integration** in the SDLC ▶ **Clarity** of the **reporting** and the **type of intervention** to be performed	▶ **Vulnerability coverage** and **few false positives** ▶ **Clarity** of the **severity** of the vulnerability and the **exploit** ▶ Verifies **exposed surface**
Effort required	▶ **Need for training** on OWASP vulnerabilities ▶ **Greater number** of potential **false positives** ▶ **Tools** to be tuned and **specific roles** in Dev teams to be definied	▶ **Medium to long-term planning** for library **updates** ▶ **Management of interdependencies** between Dev teams ▶ **SCA - Container Security** integration	▶ **Dev team training** for **DAST tool usage** ▶ **Navigation logging** for **test completeness** on developed functionalities

DevSecOps

Aspects to **address** with particular **attention** and **effort**

The Practice: which AST control for which vulnerability...

1. Develop	2. Build	3. Release	4. Deploy	5. Run
Source Code Repository	Artifact Repository	Test Environment	Staging Environment	Production Environment
SAST	**SCA**	**DAST**	**WAPT**	
2 Cryptographic Failures	6 Vulnerable and Outdated Component	2 Cryptographic Failures	1 Broken Access Control	
3 Injections	8 Software and Data Integrity Failure	3 Injections	4 Insecure Design	
7 Identification and Authentication Flaws		5 Security Misconfiguration		
9 Security Logging and Monitoring Flaws		6 Vulnerable and Outdated Component		
		7 Identification and Authentication Flaws		
		10 Server Side Request Forgery (SSRF)		

Some classes of vulnerabilities, which depend on the **business logic** and **authorization logic** of the application, might not be detectable by automatic analysis but can only be identified through **manual PTs**.

Training of teams focused on **secure development** and **security best practices** allows for effective mitigation of the risks to which applications are exposed.

Shift Left

DevSecOps

Aspects to **address** with particular **attention** and **effort**

The Practice: ... and vulns prevented by training & functional tests

Training → **Source Code Repository** → **Artifact Repository** → **Test Environment** → **Staging Environment**

0. People | **1. Develop** | **2. Build** | **3. Release** | **4. Deploy** | **5. Run**

Main Focus

1. Broken Access Control
4. Insecure Design

+

OWASP TOP 10 2021

SAST

2. Cryptographic Failures
3. Injections
7. Identification and Authentication Flaws
9. Security Logging and Monitoring Flaws

SCA

6. Vulnerable and Outdated Component
8. Software and Data Integrity Failure

DAST

2. Cryptographic Failures
3. Injections
5. Security Misconfiguration
6. Vulnerable and Outdated Component
7. Identification and Authentication Flaws
10. Server Side Request Forgery (SSRF)

"Functional Test" for security

1. Broken Access Control
4. Insecure Design

Preparing **dedicated tests** for **authorization controls** and **application logic** increases the development team's level of attention on these aspects.

SecDevOps

Aspects to **address** with particular **attention** and **effort**

The Practice: training as a "change" that must be continuously supported

Iniziative progressive di training realizzate per team di sviluppo

1. Publication of **guidelines**
2. **Intranet** section
3. **Community** (channels, blog) for interactive communication
4. **Internal webinars**
5. Online **platform** with virtual **micro-laboratories**
6. **Training courses** tailored by role (PM, SC or developer), by language, by security topic
7. **Secure Coding Academy** with annual enrollment, instructors with live webinar program, group exercises, initial and final challenge
8. **Dashboard** to verify the **security** "**score**" achieved by **individual projects** in the development phase

Aspects to **address** with particular **attention** and **effort**

The Practice: Security Champions, the key for "shift-left" & "change"

Security Champions

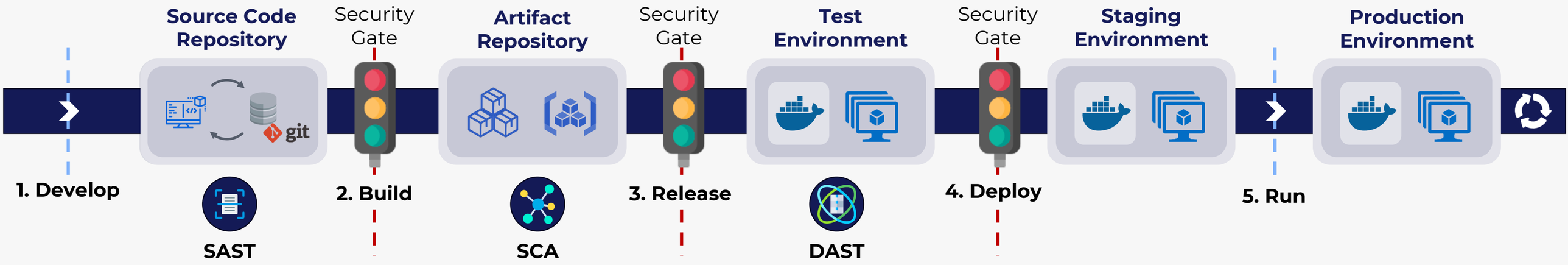
- Specialized figures integrated into development teams
- Operations limited to what is necessary
- Equipped with all necessary tools, as simple as possible

- Maximum training and support available
- Role formalization and formal designation
- Establishment of a Community registered at company level

100+ SC

Aspects to **address** with particular **attention** and **effort**

The Practice: activating Security Gates in pipelines is possible...



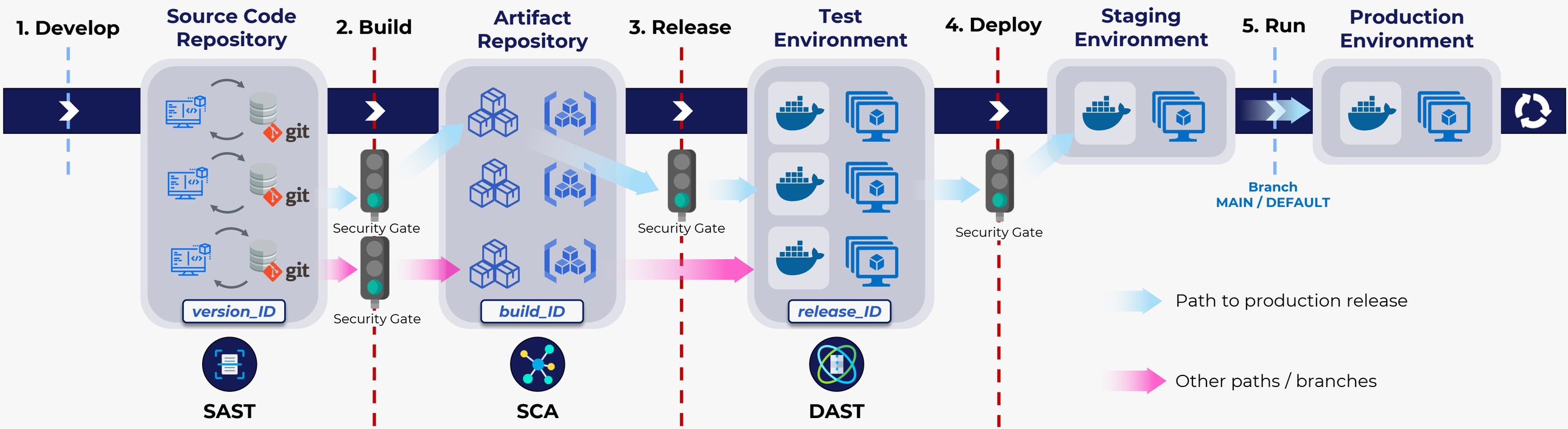
- Define for each item the identifier to be associated with the control outcome (SCA, SAST, DAST)
- Each phase must have a security gate to pass with a threshold for specific control defined uniformly with the others

Controllo	Identificativo	Oggetto del controllo	Ambito del controllo
SAST	version_ID	Source Code	GIT Repository
SCA	build_ID	Artifacts (image, .war, .jar, ...)	Artifact Repository
DAST	release_ID	URL, credentials, test data	Test Environment

SecDevOps

Aspects to **address** with particular **attention** and **effort**

The Practice: ... but in which branch?!

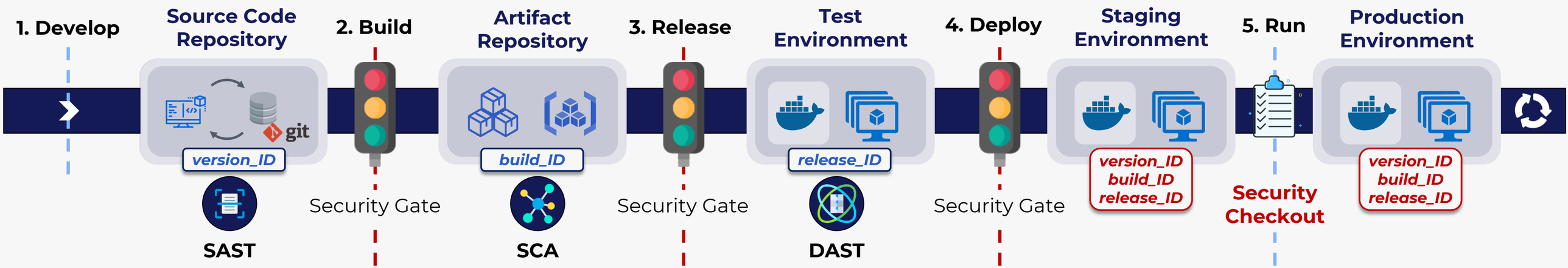


- There must be a **single path** to reach **production** (e.g. MAIN/DEFAULT Branch)
- **Security Gates *CAN*** be **set** on **any path**, but **MUST** be **present and active** on the path for **production release**
- **Definition**, **monitoring** and **enforcing** of **guidelines** and **corporate rules**

SecDevOps

Aspects to **address** with particular **attention** and **effort**

The Practice: ... but more complex to implement Security Check-out!



▶ **Objective**: to be **certain** that what reaches **production** has passed **ALL security controls** through the implementation of a **Security Checkout**

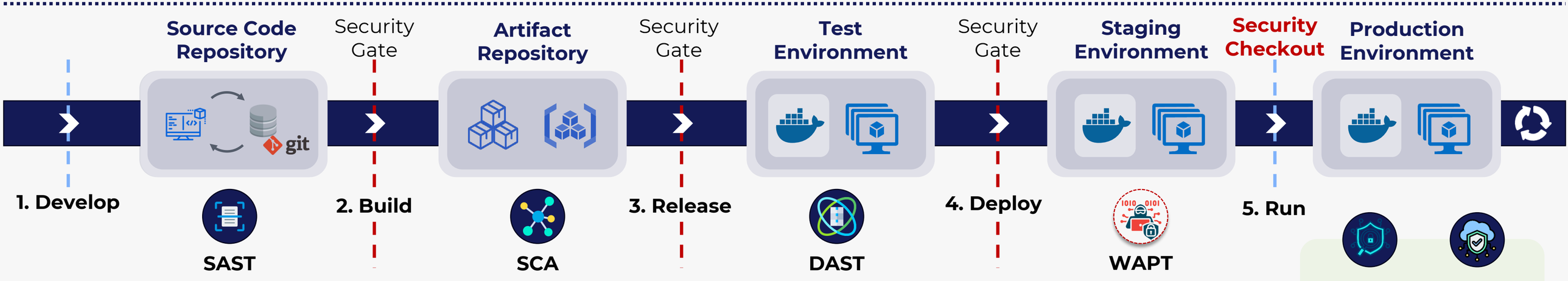
To develop the **Security Checkout** it is necessary to **centralize and register**:

- **Identifiers** of all **sw components** (version, build and release ID)
- **Outcome** and **timestamp** of all **controls performed**

The **Security Checkout** verifies that **for each of the sw components** in the **production release** phase, **all AST security controls associated** with the respective **identifiers are passed**

SecDevOps

The Theory: ... and then shift right



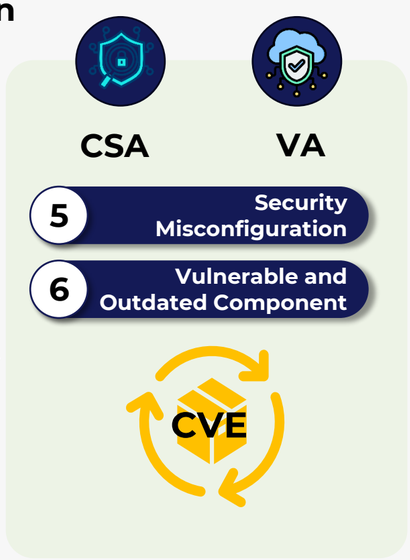
After passing the **Security Checkout**, what security **problems** can manifest at **runtime**?

- **New vulnerabilities** in libraries and products
- **Misconfigurations** introduced during operation

Periodic controls to activate at **runtime**:

- **Vulnerability Assessment** on servers
- **Container Security Analysis** on microservices

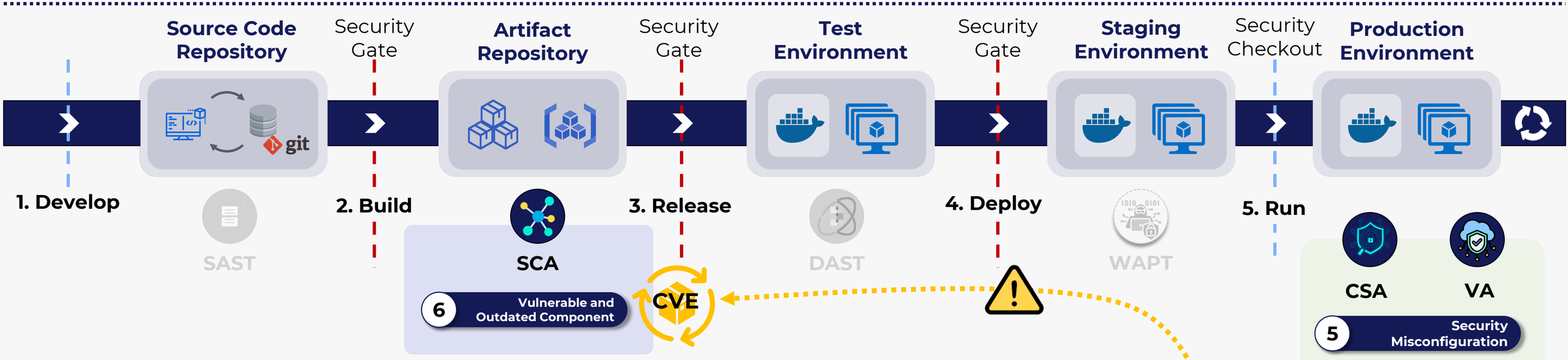
Presence of CVEs and vulnerable configurations in systems, libraries and products used.

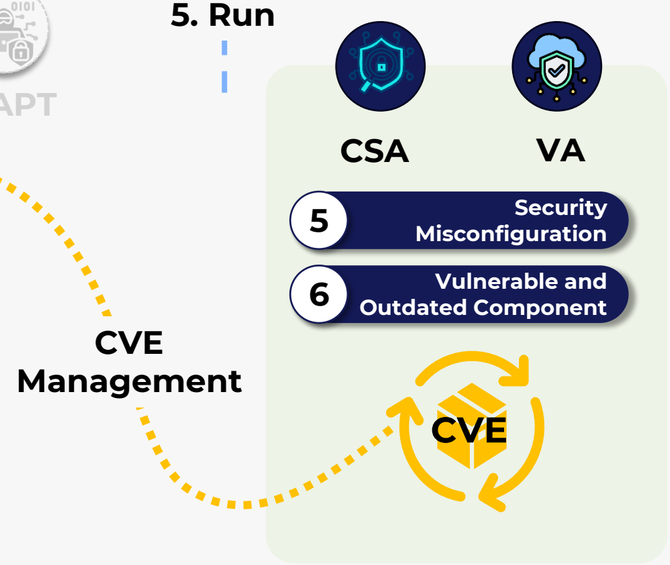


SecDevOps

Aspects to **address** with particular **attention** and **effort**

The Practice: ... and then shift back again





SAST, DAST, WAPT	SCA, VA, Container Security
Code fixing	CVE management
Linear process (fix it once)	Cyclic process (continuous monitor & update/upgrade)
Check in Dev	Check in Dev & Ops
Short-term fix	Short-medium-long term plan

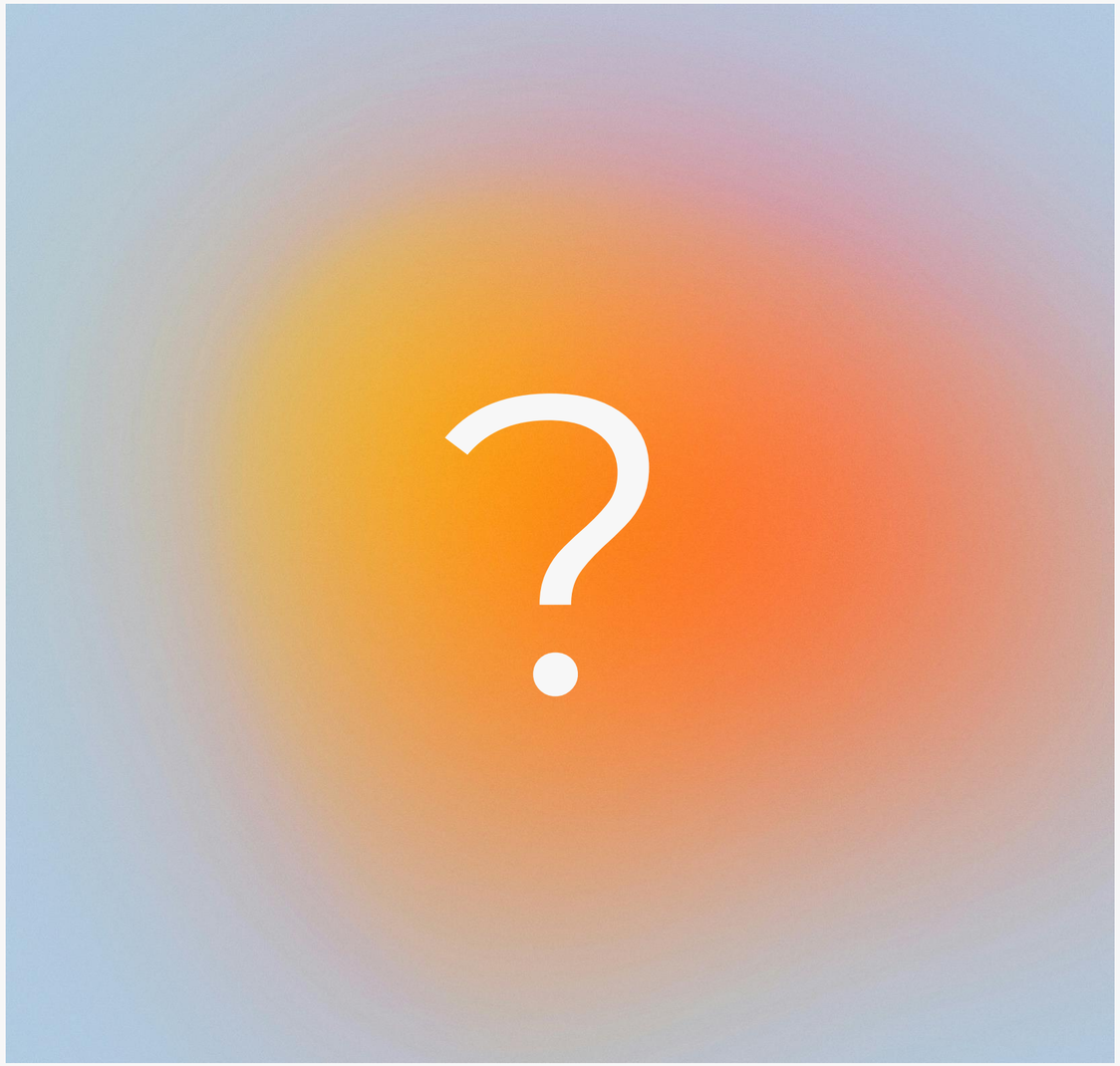
SecDevOps

Conclusions: ... shift focus to people

Improving the security level of source code does not require only a **technological intervention** to insert automatic AST controls in the SDLC, but also an **organizational process** to involve the necessary **people**, to be addressed by applying **Change Management strategies** especially in large organizations:

- ✓ **Clarity of objectives, tasks and benefits** for participants,
- ✓ **Essential effort and gradual path**,
- ✓ **Bidirectional communication/collaboration**,
- ✓ **Community & Champions**,
- ✓ **"Empathy" and continuous support**,
- ✓ **Corporate commitment.**

Q&A Session



GRAZIE PER L'ATTENZIONE

OWASP 2025 ITALY DAY

19th JUNE 2025

FRONTEMARE CAGLIARI - SARDINIA