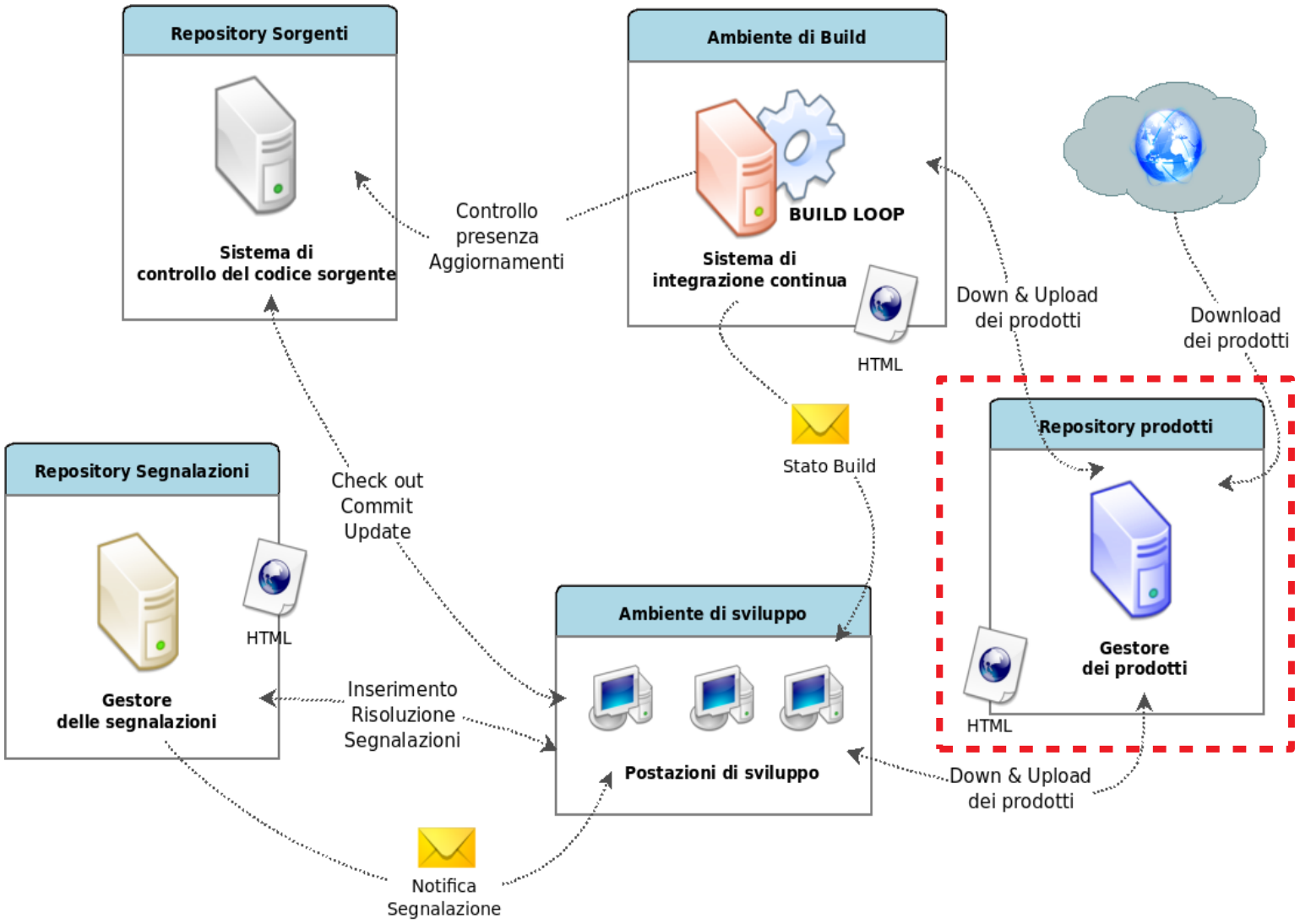
Artifact Repository

METODI E TECNOLOGIE PER LO SVILUPPO SOFTWARE

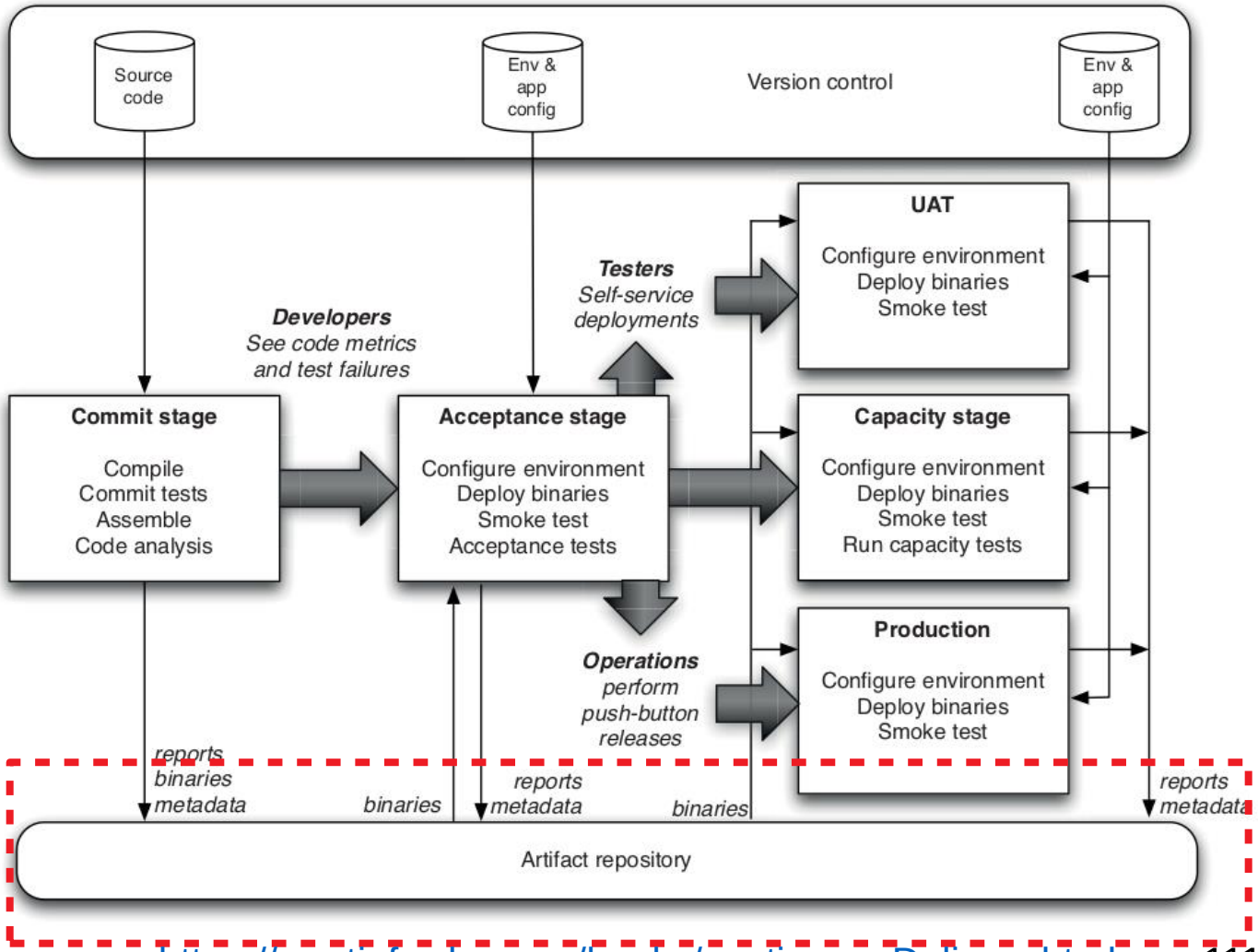
Nicola Bertazzo
nicola.bertazzo [at] unipd.it

Università degli Studi di Padova
Dipartimento di Matematica
Corso di Laurea in Informatica, A.A. 2022 – 2023

Visione Generale



Visione Generale



https://martinfowler.com/books/continuousDelivery.html pag 111

Binary repository manager

Definizione

A **binary repository manager** is a software tool designed to **optimize** the **download** and **storage of binary files used** and **produced** in **software development**. It centralizes the management of all the binary **artifacts generated** and **used** by the organization to overcome the complexity arising from the diversity of binary artifact types, their position in the overall workflow and the dependencies between them.

A binary repository is a software repository for **packages**, **artifacts** and **their corresponding metadata**. It can be used to store **binary files produced** by an organization itself, such as product releases and nightly product builds, **or for third party binaries** which must be **treated differently for both technical and legal reasons**.

WIKIPEDIA
The Free Encyclopedia

Definizione 1/2

The outputs of the commit stage, your reports and **binaries**, **need to be stored somewhere for reuse in the later stages of your pipeline**, and for your team to be able to get hold of them. The obvious place might appear to be your version control system. There are several reasons why this is not the right thing to do, apart from the incidental facts that in this way you're likely to work through disk space fast, and that some version control systems won't support such behavior.

https://martinfowler.com/books/continuousDelivery.html

Definizione 2/2

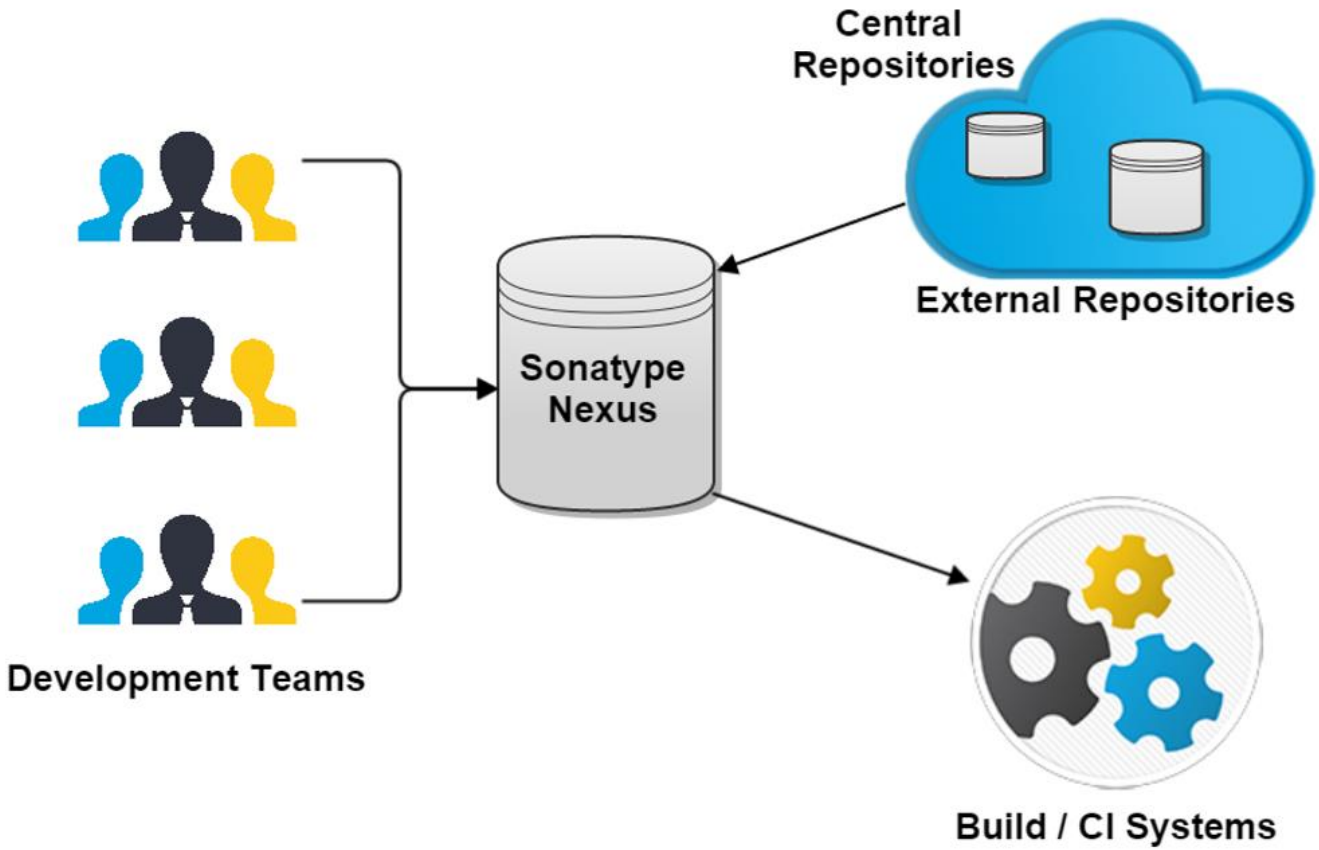
The artifact repository is an unusual kind of version control system, in that it **only needs to keep some versions**. Once a **release candidate has faile some stage in the deployment pipeline, we are no longer interested in it**. So we can, if we wish, purge the binaries and reports from the artifac repository.

It is essential to be able to trace back **from your released software to the revisions in version control that were used to create it**. In order to do this an instance of a pipeline should be correlated with the revisions in your version control system that triggered it. **Checking anything into source control as part of your pipeline makes this process significantly more complex by introducing further revisions associated with your pipeline.**

One of the acceptance criteria for a good configuration management strategy is that the binary creation process should be repeatable. That is, **if I delete the binaries and then rerun the commit stage from the same revision** that originally triggered it, **I should get exactly the same binaries again.** Binaries are second-class citizens in the world of configuration management, although it is worth keeping hashes of your binaries in permanent storage to verify that you can re-create exactly the same thing and to audit back from production to the commit stage.

https://martinfowler.com/books/continuousDelivery.html

Artifact Repository



https://cloudogu.com/en/blog/scm-manager-universe-tools-part-4-sonatype-nexus

Artifact Repository

Caratteristiche

Ambiente dove depositare e pubblicare i prodotti

Agisce da intermediario per scaricare prodotti da depositi esterni

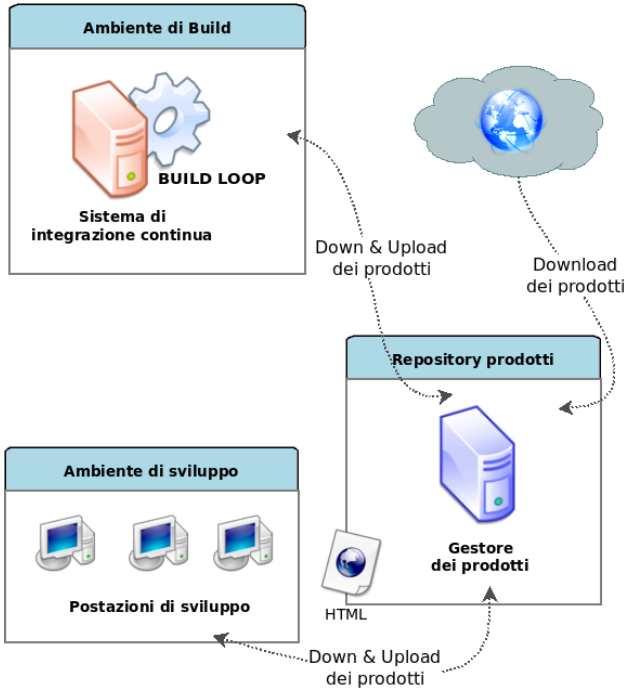
Permette di effettuare ricerche e reperire informazioni riguardanti i prodotti

Permette di gestire e associare permessi d'accesso sui prodotti

Permette di segnalare problemi di vulnerabilità su prodotti

Permette di verificare problemi legati a licenze (dipendenze con prodotti terzi)

Permette di documentare gli artefatti con dei metadati (p.es pom.xml)



Artifact Repository

Tipi di artefatti

Caratteristiche degli artefatti

Possibilità di distinguere univocamente un artefatto (G.A.V)

Metadati che permettono di capire se (p. es. pom.xml):

- Artefatto di produzione/sviluppo (politiche di gestioni differenti)
- Riferimento del commit nel VCS da cui è stato creato l'artefatto
- Informazioni delle licenze
- Informazioni delle dipendenze
- MD5, SHA1 per garantire l'autenticità dell'artefatto
- Identificativo univoco

Artifact Repository

Maven Repository Management

Proxy Remote Repositories: Possibilità di configurare il repository interno in modo da recuperare gli artefatti da repository esterni. Se l'artefatto non è presente nel repository interno viene consultato il repository esterno

Hosted Internal Repositories: Possibilità di condividere all'interno di un'organizzazione artefatti di terze parti che hanno vincoli di licenza (p.es. Dirver JDBC)

Release Artifacts: Gli artefatti di tipo Release solitamente possono essere rilasciati una sola volta e vengono mantenuti nel repository.

Snapshot Artifacts: Gli artefatti in fase di sviluppo possono essere rilasciati più volte. Possono essere eliminati (e mantenuta la versione più recente). Di solito contengono il la Keyworkd SNAPSHOT e il timestamp

https://dzone.com/refcardz/getting-started-repository?chapter=1#refcard-download-social-buttons-display

Artifact Repository

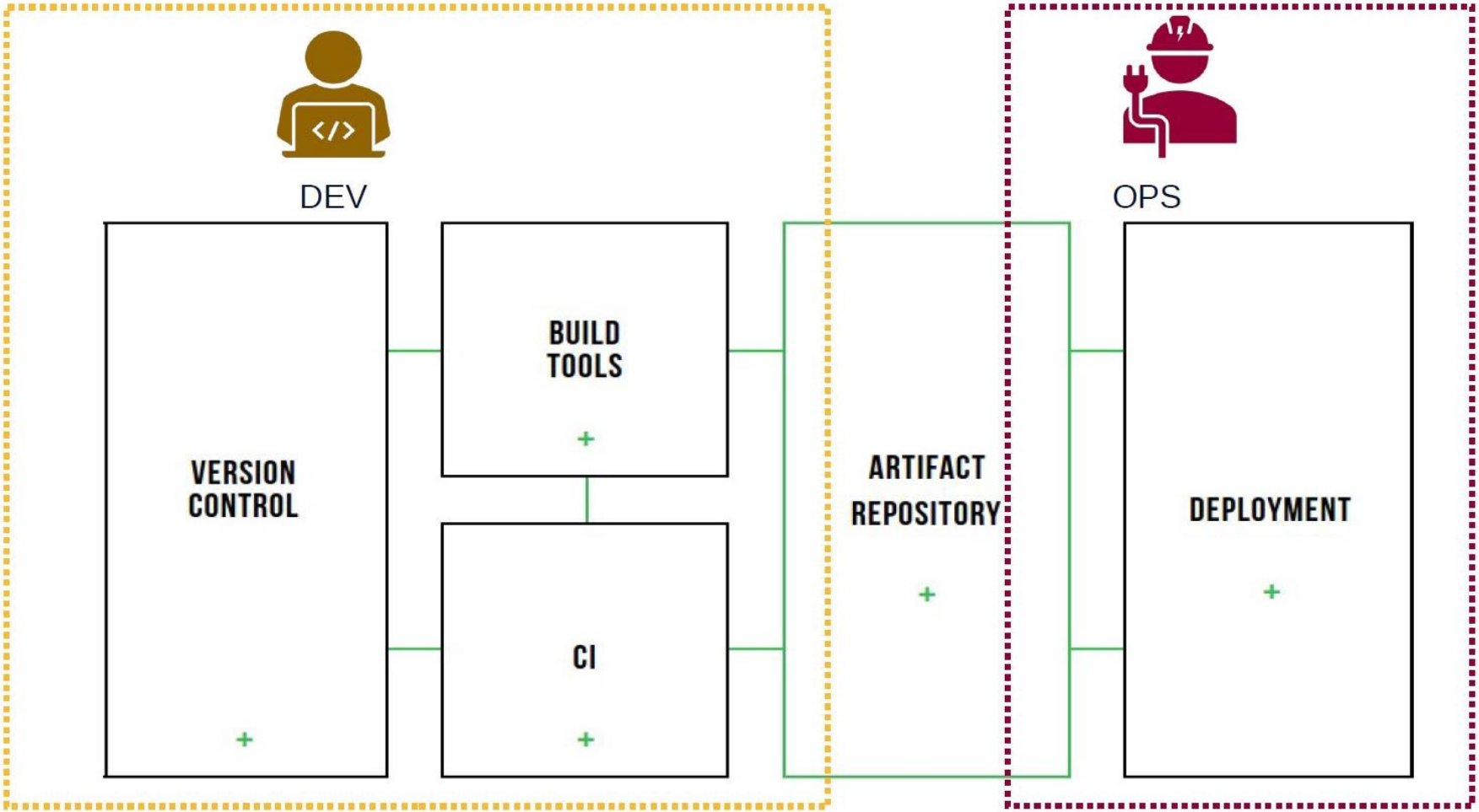
Perché usare un Maven Repository Interno

Velocità del processo di Build: Le dipendenze e gli artefatti di tipo plugin utilizzati nel processo di build vengono scaricati dalla rete interna

Maggiore stabilità e controllo: I repository gestiti possono essere analizzati e si può decidere di vietare l'utilizzo di determinati artefatti per evitare problemi di sicurezza. È possibile distribuire le versioni ufficiali e certificate degli artefatti di terze parti. È più semplice fare analisi (p.es. Compatibilità delle licenze del prodotto con le dipendenze)

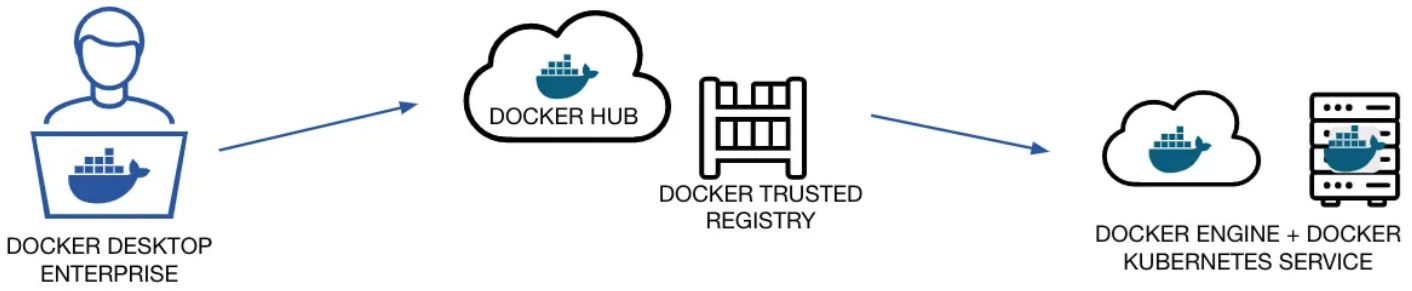
Facilitano la collaborazione: Si evita di far creare gli artefatti dal VCS ed è più semplice identificare le versioni di rilascio certificate.

https://dzone.com/refcardz/getting-started-repository?chapter=1#refcard-download-social-buttons-display


DEV
OPS
VERSION CONTROL
BUILD TOOLS
CI
ARTIFACT REPOSITORY
DEPLOYMENT

Artifact Repository

Caso d'uso docker



BUILD:

- Define and package multiple images and their interdependencies
- Compatible with Docker Compose, Helm charts and Kubernetes YAML

SHARE:

- Collaborate and distribute via Docker Hub and Docker Trusted Registry
- Shareable applications with clear interfaces for operators

RUN:

- Run multiple versions of the same application and manage per-environment settings
- Works with Swarm and Kubernetes

https://www.docker.com/blog/multi-service-applications-with-docker-enterprise-3-0/

Caso d'uso: Maven Gitflow

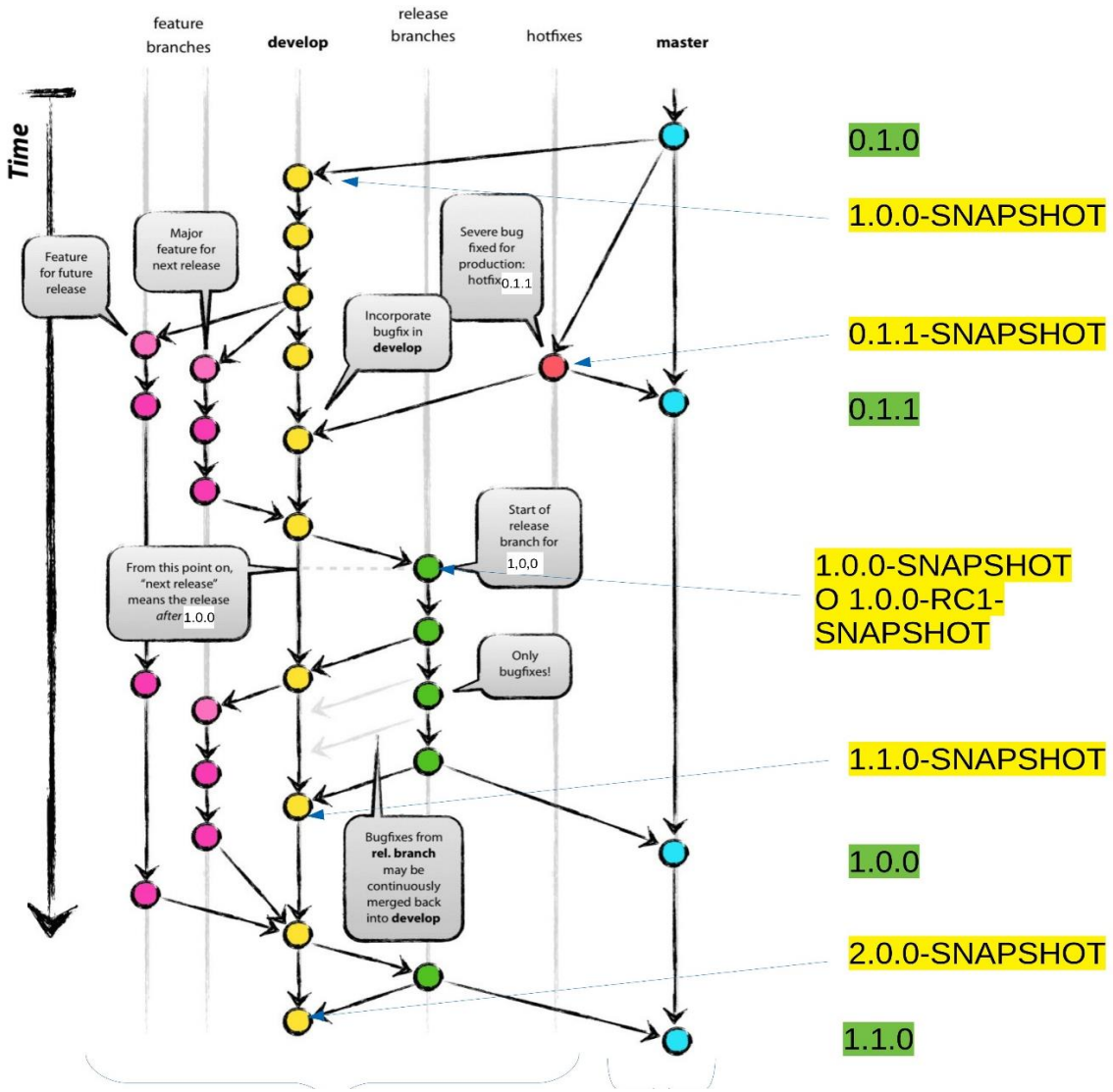
Repository Snapshot

- Suffisso SNAPSHOT
- Posso pubblicare la versione più volte (nel repository viene registrato anche il timestamp di pubblicazione

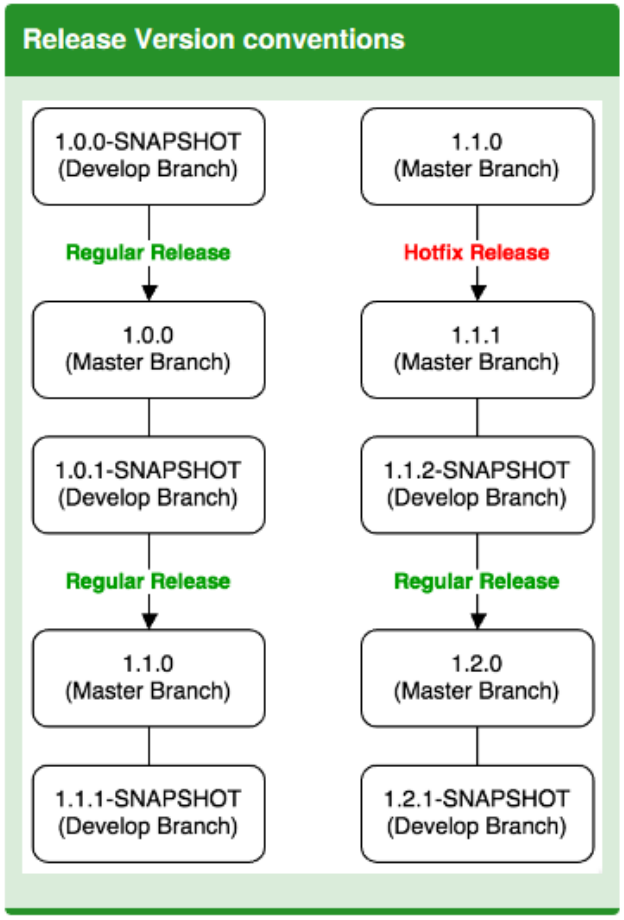
Repository Release

- Posso pubblicare la versione solo 1 volta.
- No dipendenze e moduli di tipo SNAPSHOT

Artifact Repository



Caso d'uso: Maven Gitflow



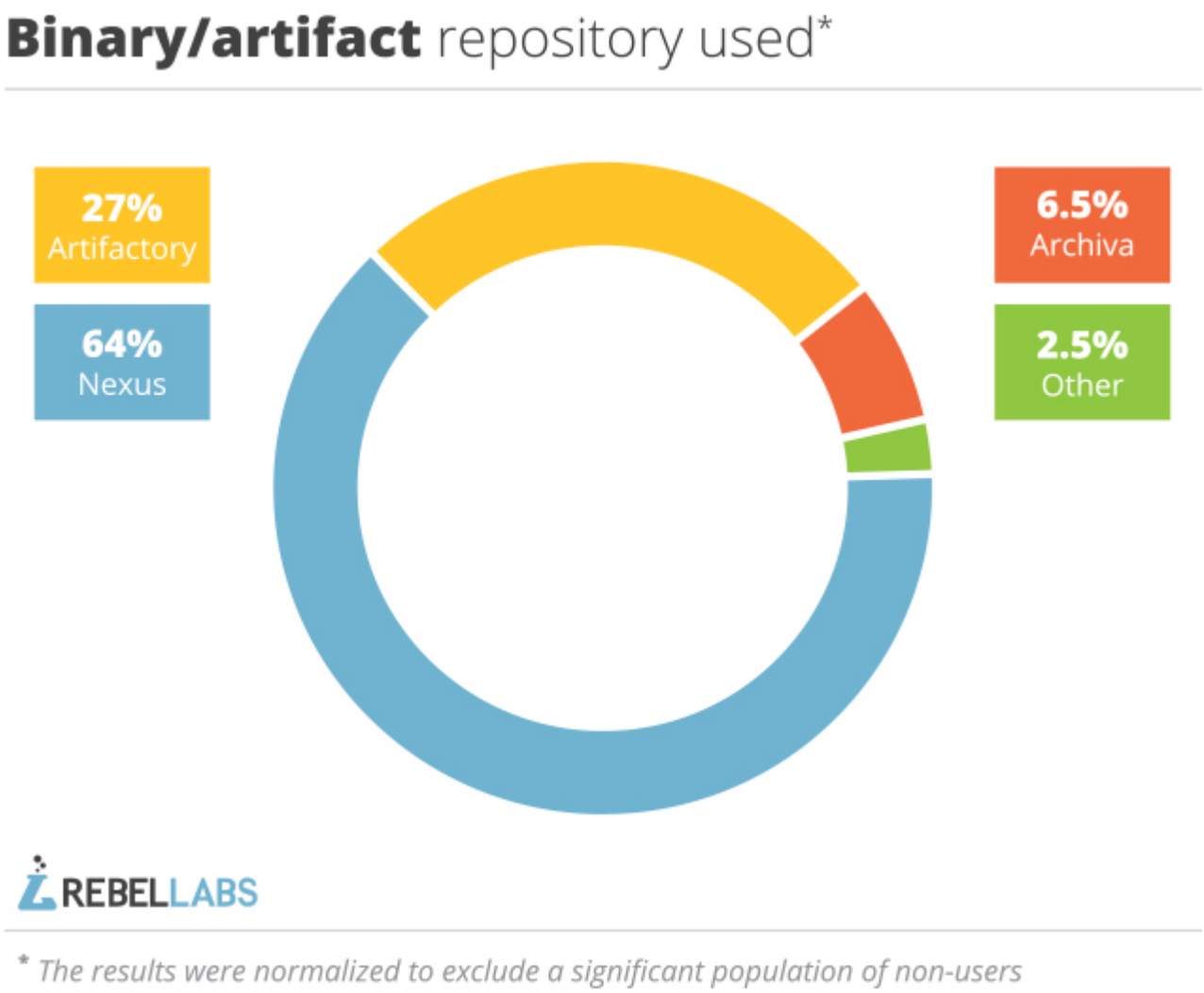
https://www.cheatography.com/mikesac/cheat-sheets/gitflow/

https://blog.sonatype.com/2015/08/automated-nexus-reports-on-licenses-security-and-more/

https://www.jfrog.com/confluence/display/RTF/License+Control

https://jfrog.com/xray/

Artifact Repository



Riferimenti

https://en.wikipedia.org/wiki/Binary_repository_manager
https://en.wikipedia.org/wiki/Binary_repository_manager
https://zeroturnaround.com/rebellabs/java-tools-and-technologies-landscape-for-2014/15/
https://cdn2.hubspot.net/hubfs/1958393/eBooks/AHC_Guide.pdf
https://blog.sonatype.com/2009/04/what-is-a-repository/
https://jfrog.com/artifactory/
https://www.sonatype.com/nexus-repository-sonatype
https://www.sonatype.com/hubfs/DevSecOps%20Reference%20Architectures%202018.pdf
https://cloudogu.com/en/blog/scm-manager-universe-tools-part-4-sonatype-nexus
https://dzone.com/refcardz/getting-started-repository?chapter=1#refcard-download-social-buttons-display
https://blog.sonatype.com/2009/04/what-is-a-repository/

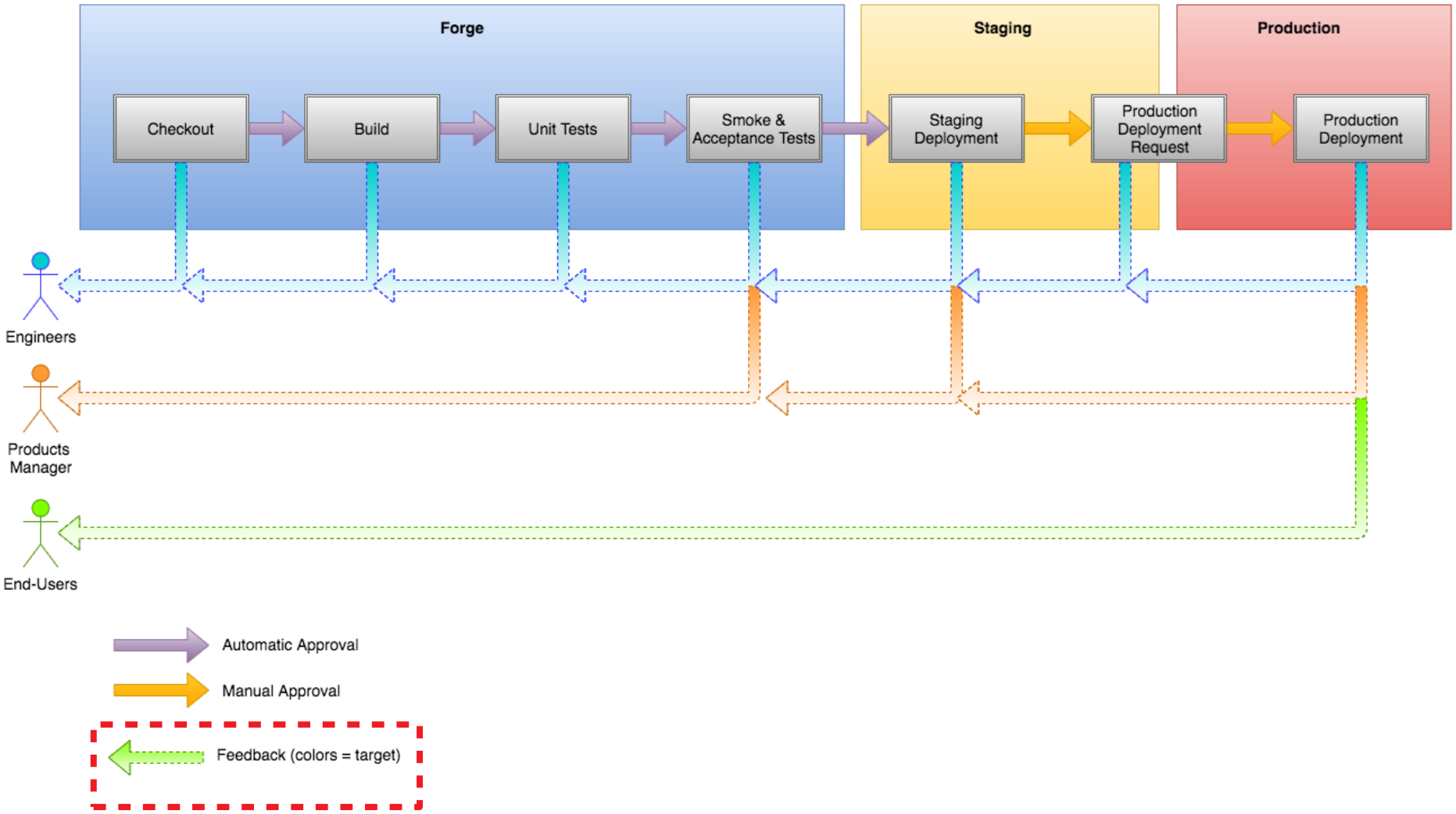
Feedback looop

METODI E TECNOLOGIE PER LO SVILUPPO SOFTWARE

Nicola Bertazzo
nicola.bertazzo [at] unipd.it

Università degli Studi di Padova
Dipartimento di Matematica
Corso di Laurea in Informatica, A.A. 2022 – 2023

Visione Generale



Contesto

Contesto: Continuous Delivery / Deployment Pipeline
Obiettivo: FeedBack veloci per reagire velocemente

Necessità di sistemi che permettono di inviare velocemente feedback a tutti gli attori coinvolti nel processo di Deploy

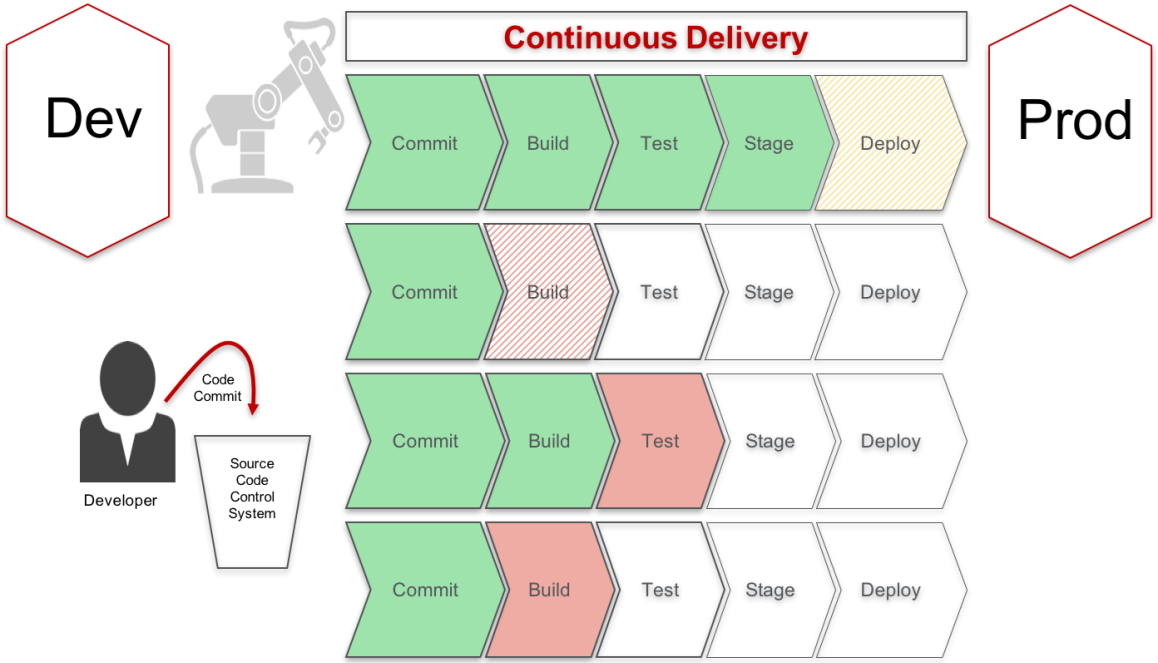
- **Efficienza** è la chiave del successo
- **Confidenza**: Tenere informati gli attori quando si rompe qualcosa e quando ritorna la normalità
- **Attendibilità**: inviare messaggi corretti, che danno un valore aggiunto

Feedback Loop

Ces'è un Feedback loop

Il feedback loop è l'insieme di strumenti che:

- Forniscono **lo stato di ogni passo** (stage) presente nella pipeline
- Inviano il **messaggio giusto** al persona giusta
- Permettono di **misurare** il processo



Feedback Loop

Come implementare un feedback loop?

Valutare:

- **Quali attori** notificare? (Developers, Operation, Project Manager, Business)
- **Che media** utilizzare? (Direct Visualization, E-Mail, IRC, Collaboration chats, Video messaging, Extreme Feedback Devices)
- **Che evento**, **priorità**, **frequenza**, limiti? (modifica dello stato di un Stage, Mail gray-listing, Production feedback vs. testing feedback)

Vedi: https://www.cloudbees.com/blog/sending-notifications-pipeline