IUT
Belfort-
Montbéliard

Tests d'intégration & Intégration continue

INFO
Département Informatique
IUT Belfort-Montbéliard

UNIVERSITÉ DE FRANCHE-COMTÉ

Définitions

Issues du glossaire CFTL/ISTQB des termes utilisés en tests de logiciels

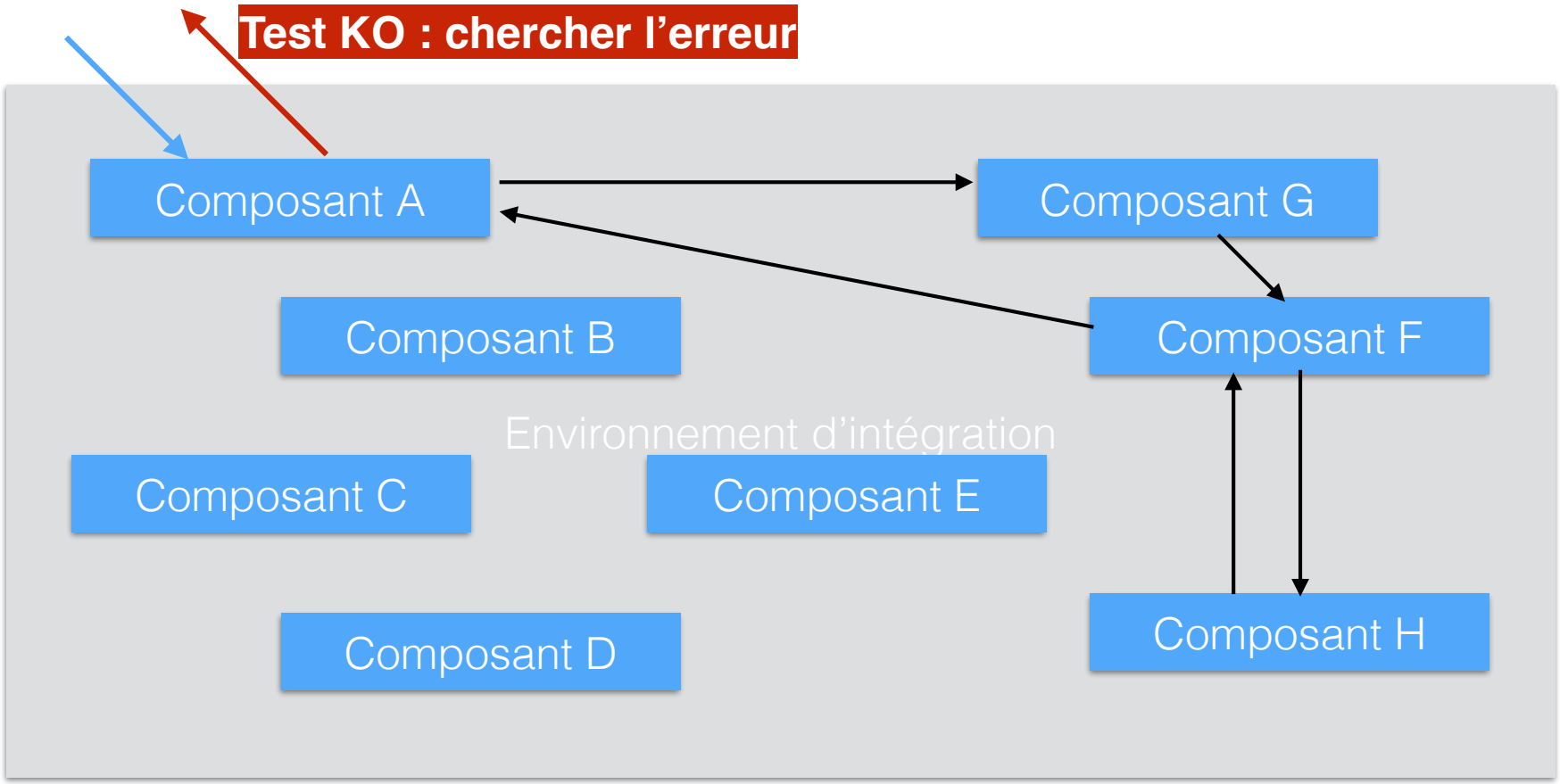
- Intégration : le processus de combiner des composants ou systèmes en assemblages plus grands.

- Tests d'intégrations : tests effectués pour montrer des défauts dans les interfaces et interactions de composants ou systèmes intégrés.

Stratégies d'intégration

- Dans quel ordre considérer les composants à intégrer ?
  - en fonction du planning des livraisons
  - en fonction des interfaces et interactions entre les composants
  - en fonction des flux (métiers ou techniques) sollicitant plusieurs composants

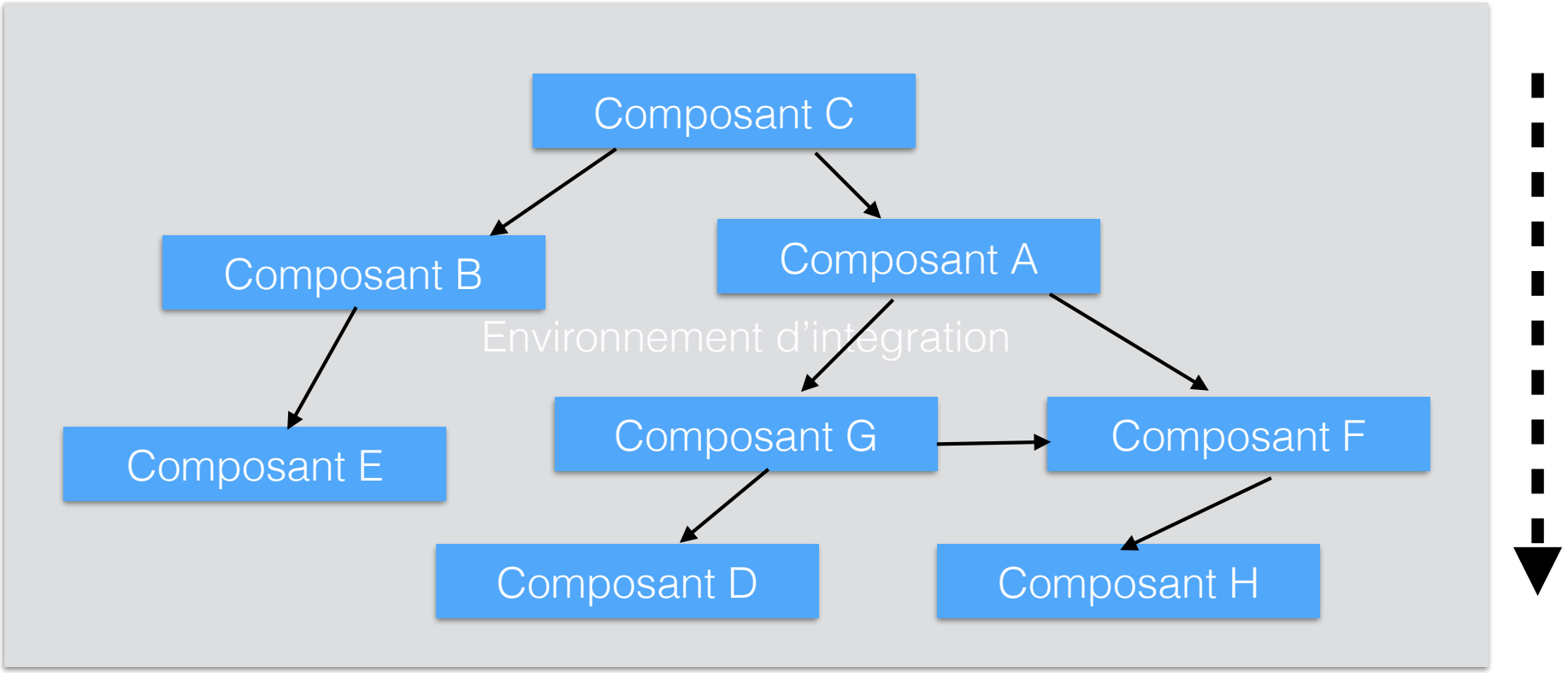
Intégration Big-Bang



Tous les composants sont intégrés en même temps

- Tous les composants sont intégrés simultanément
- Il faut donc attendre d'en disposer
- La recherche d'une erreur est fastidieuse
- À éviter

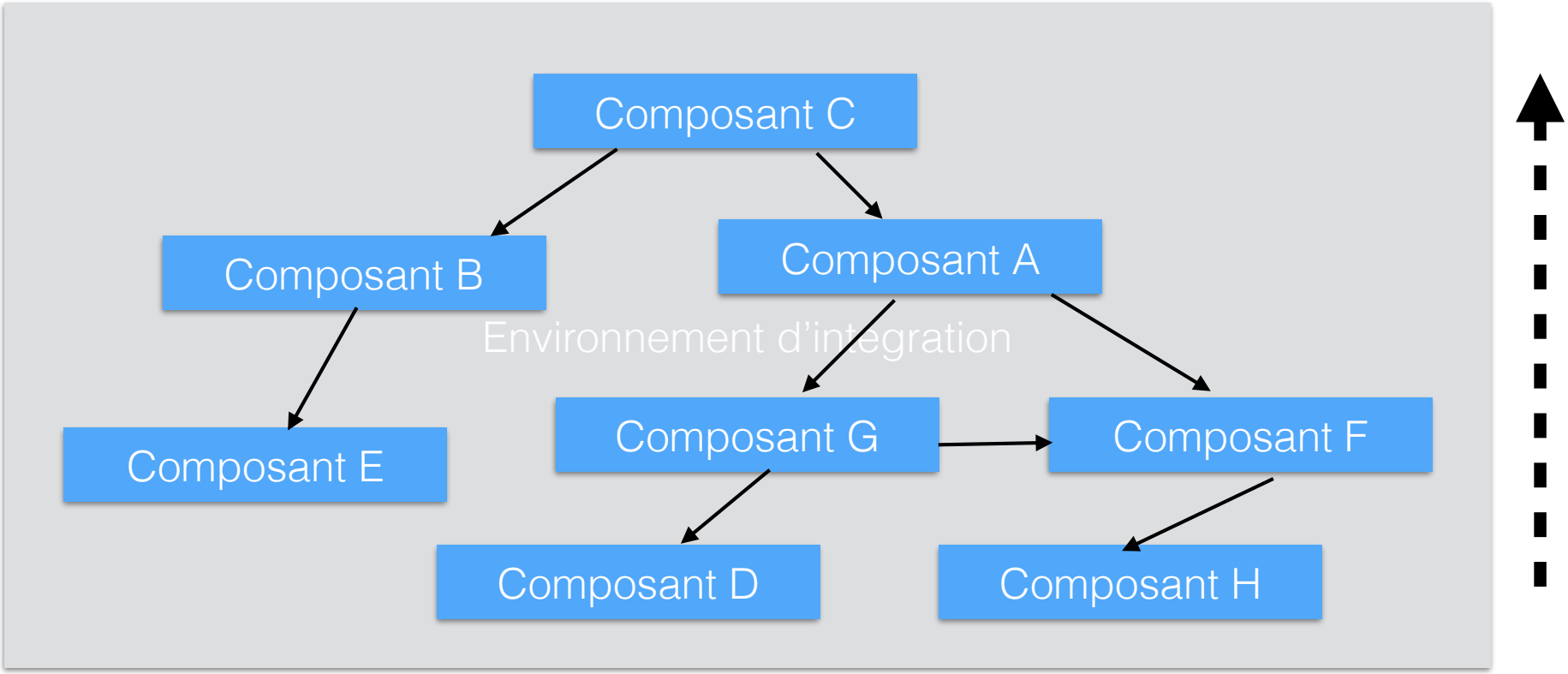
Intégration Top-Down



- Intégrer les composants en commençant par le moins sollicité puis en descendant le graphe de dépendance

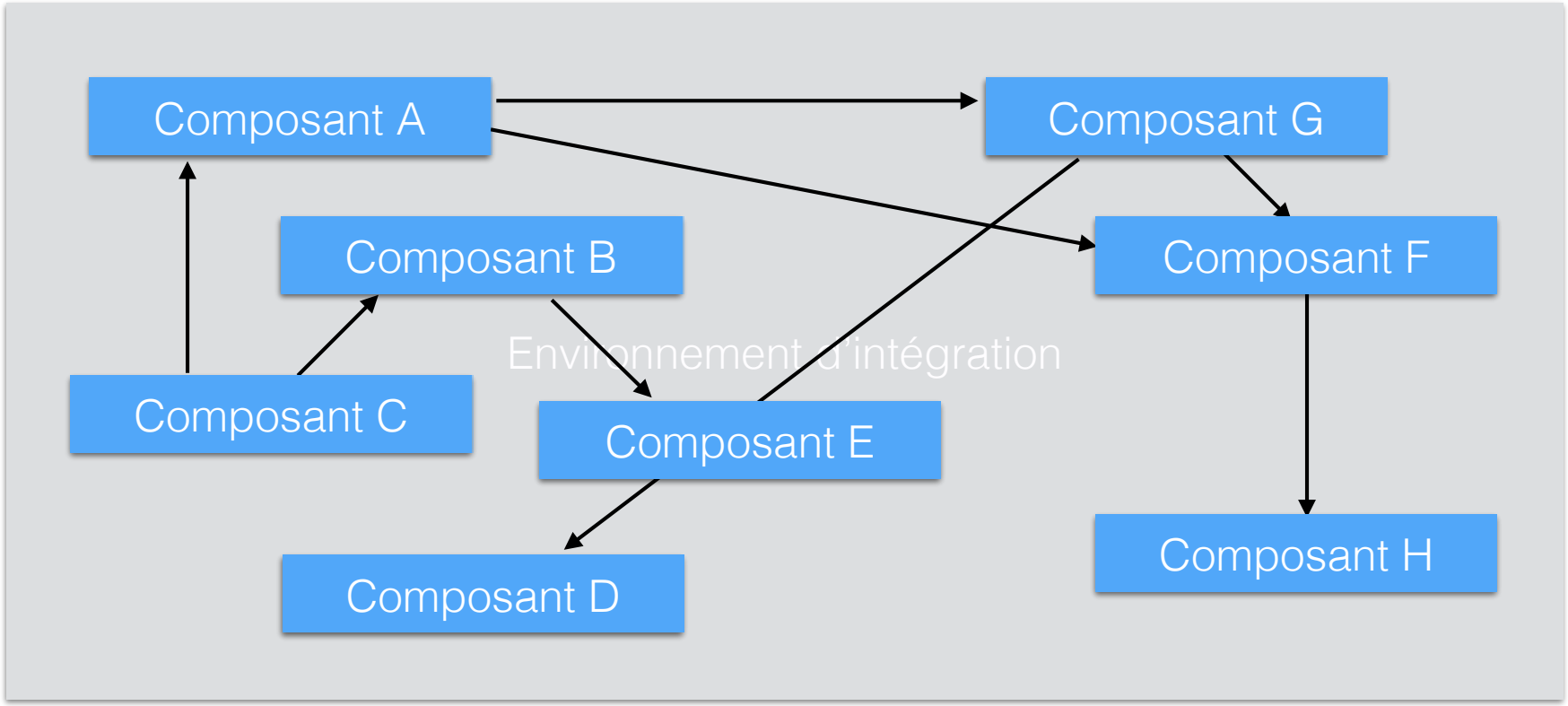
- L'intégration se fait progressivement et nécessite l'écriture de bouchons (pour les composants de niveau inférieur non encore intégrés)

Intégration Bottom-UP



- Des composants feuilles vers les composants appelants
- Nécessite l'écriture de pilotes pour les composants de niveau supérieur non encore intégrés

Intégration Continue



- L'architecture d'intégration est anticipée en construisant les pilotes et bouchons des différents composants
- Dès qu'un composant est disponible il est intégré dans l'architecture
- Nécessite la définition préalable des interfaces entre composants

Pré-requis pour une Intégration Continue réussie

- Partage du code source (logiciel de gestion de version)
- Dépôts (commit) quotidien (mini) des modifications
- Tests d'intégration développés
- Compilations et constructions des livrables automatisées
- Mise en œuvre d'un outil d'intégration continu

Étapes d'intégration continue automatisée

Étape 1 : mise à jour des sources

Serveur de déploiement

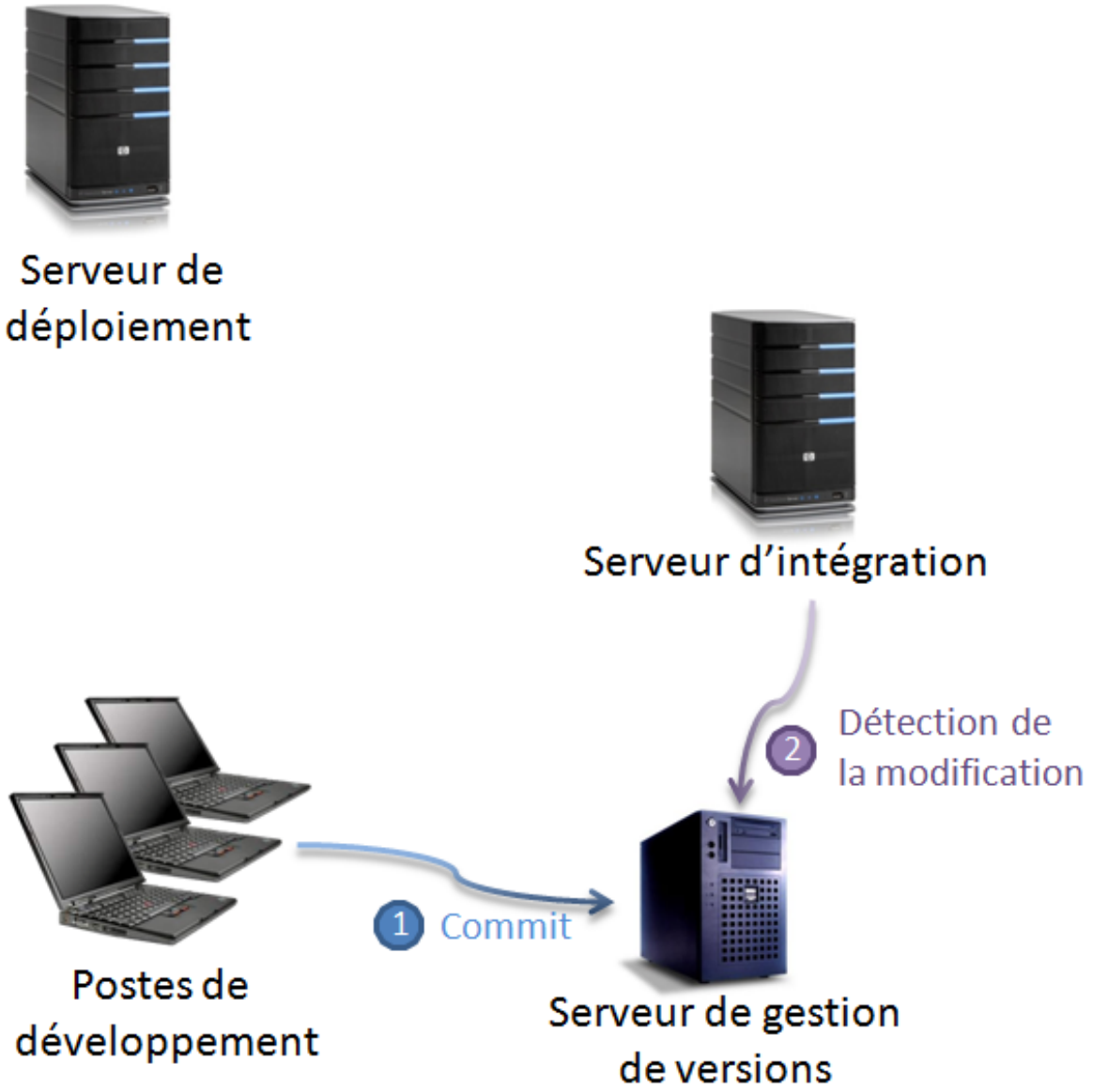
Serveur d'intégration

Postes de développement

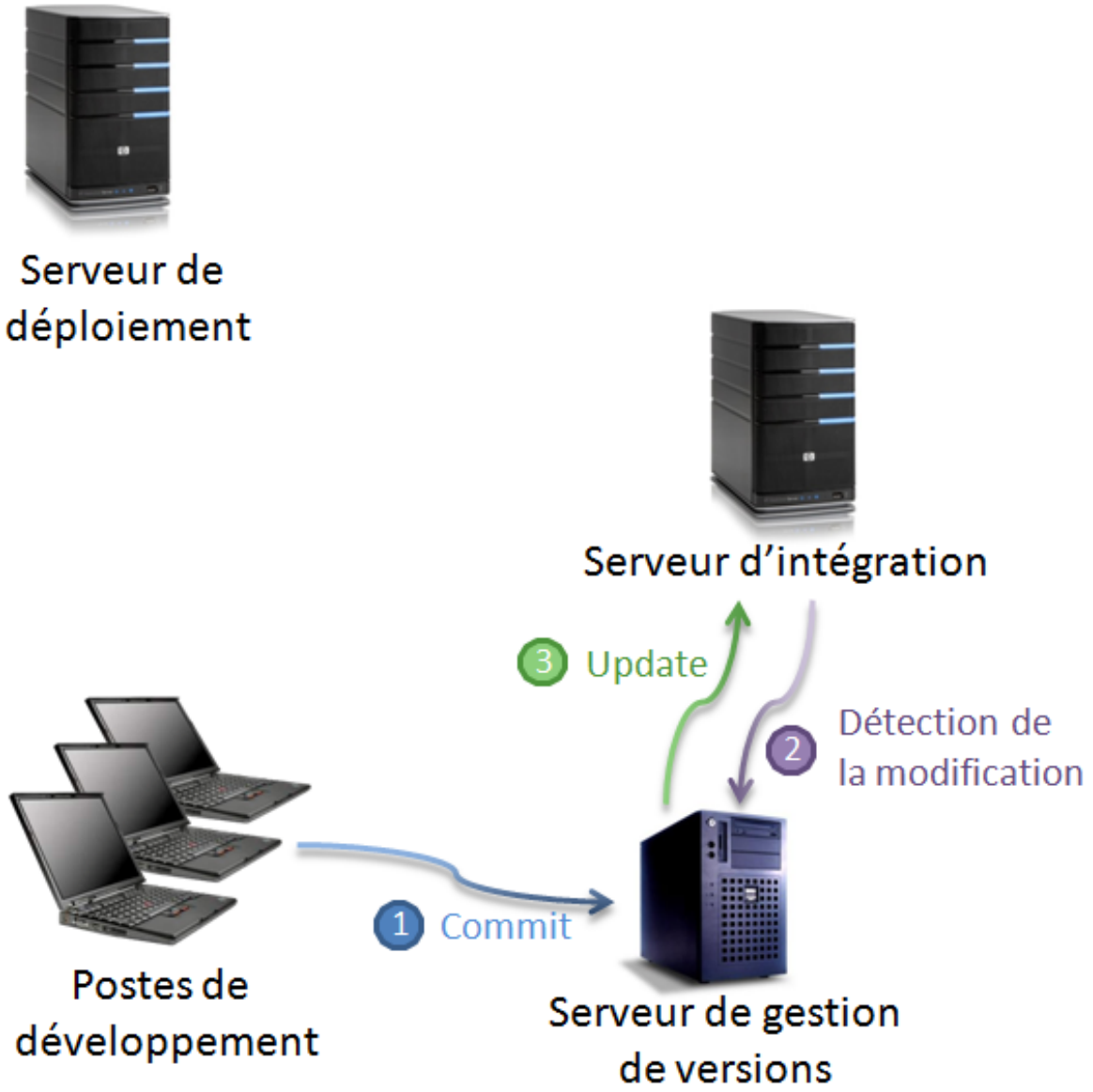
1 Commit

Serveur de gestion de versions

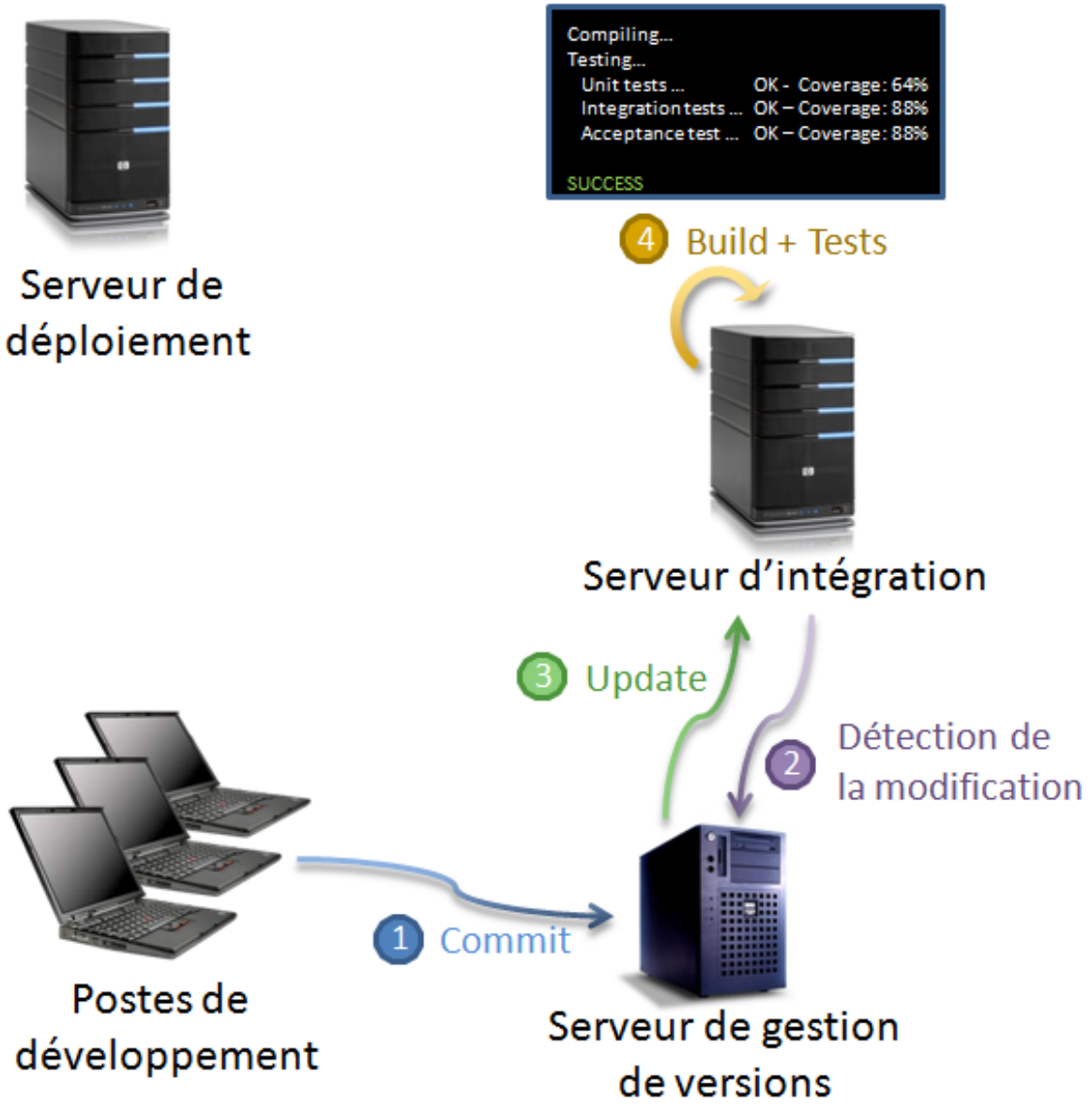
Étape 2 : détection de la mise à jour



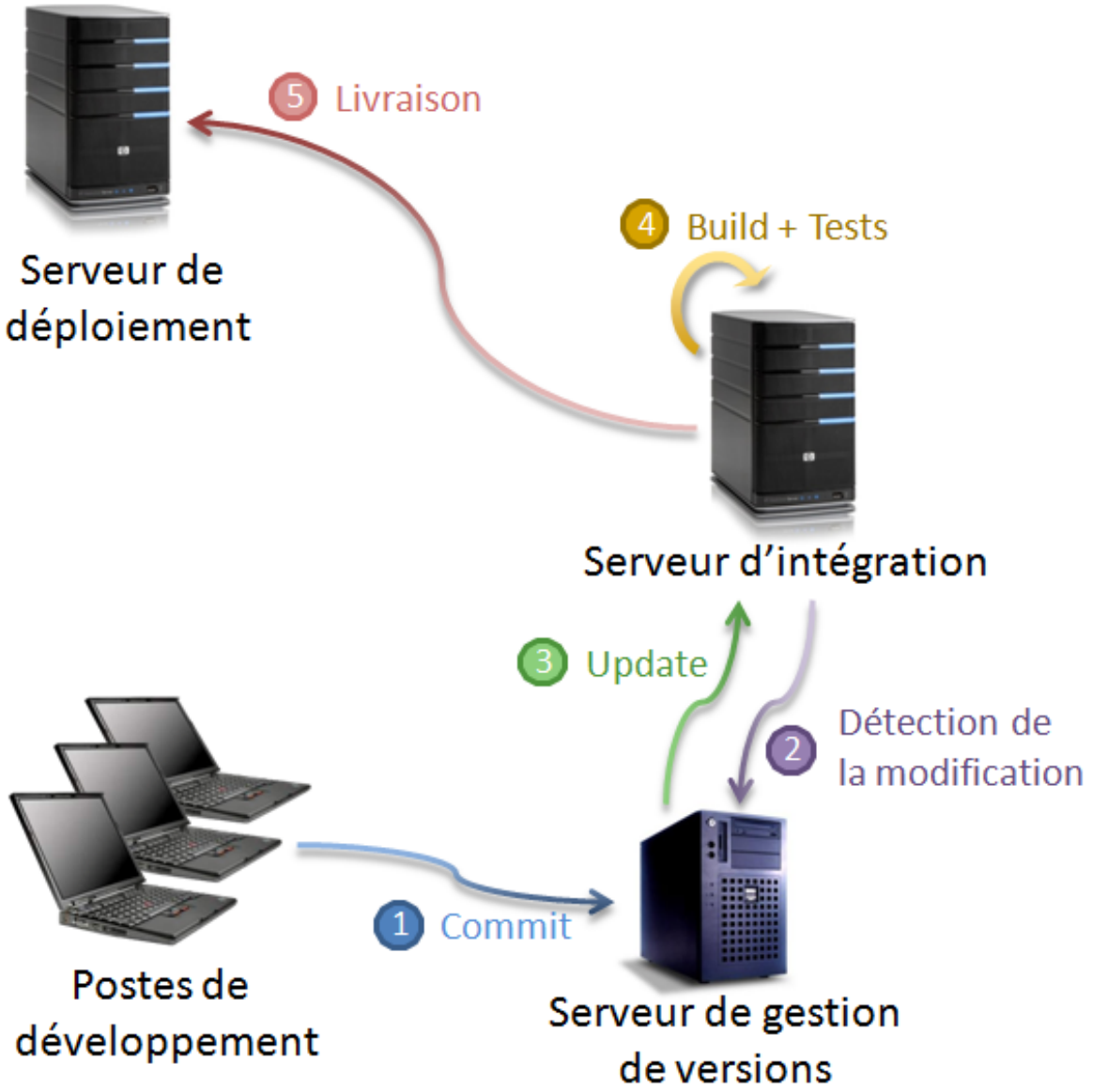
Étape 3 : récupération des sources



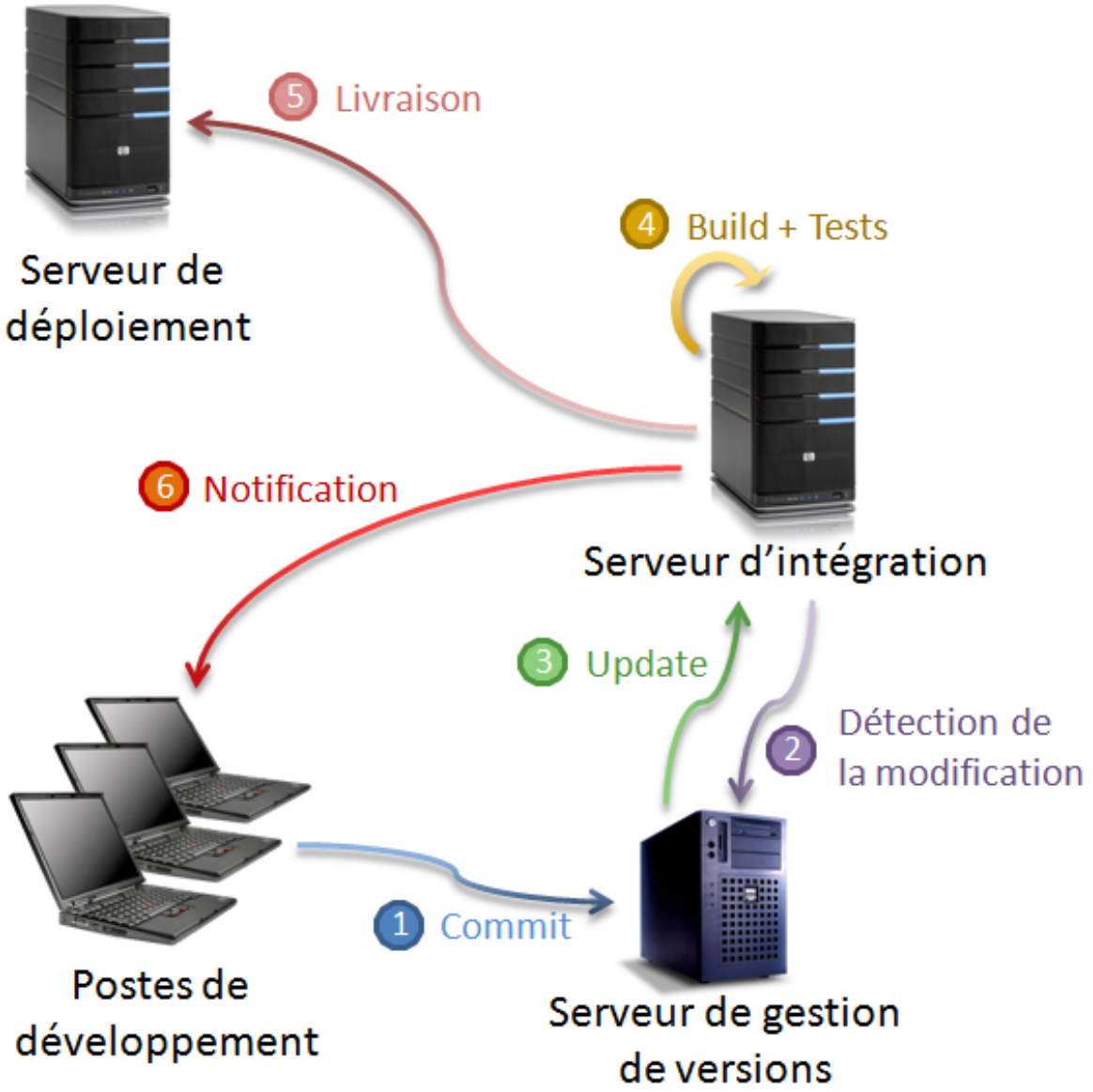
Étape 4 : construction et tests release



Étape 5 : livraison de la release



Étape 6 : publication du rapport



Outils d'intégration continue

Jenkins

Hudson

continuum

cruisecontrol

Bamboo

Automatiser la compilation et la construction des livrables

Outils

Maven™

Gradle

Maven

- Basé sur POM (Project Object Model) et de nombreux plugins
- Impose une structure au projet
- Gère les dépendances lors de la construction des artefacts <build/>
- Permet la production de rapports html <reporting/>

Architecture

- projet/ : dossier du projet
  - pom.xml : fichier de configuration Maven
  - src/ : dossier des sources
    - main/ : contient les sources de l'application
      - java/ : code source java
      - resources/ : ressources associées au code
    - test/ : contient les tests de l'application
      - java/ : code source des tests
      - resources/ : ressources associées au tests
  - target : dossier contenant les résultats, les binaires et toutes les sorties de build Maven

Commande

- mvn ***cible***
- ***cible*** :
  - compile : compile le projet
  - test : exécute les tests
  - package : crée le fichier .jar de votre programme
  - install : installe le .jar dans le dépôt local
  - deploy : déploie le .jar sur un dépôt distant
  - clean : supprime les résultats des commandes précédentes

pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
          http://maven.apache.org/xsd/maven-4.0.0.xsd">

    <modelVersion>4.0.0</modelVersion>
    <groupId>MyGroup</groupId>
    <artifactId>MyArtifact</artifactId>
    <version>1.0-SNAPSHOT</version>
</project>
```

mentions obligatoires : groupId, artifactId, version

Les versions

- X.x-SNAPSHOT : version de travail, peut être déployée plusieurs fois
- X.x : release, unique, ne peut pas être déployée plusieurs fois

Les dépendances

```
<dependencies>
    <dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>4.12</version>
        <scope>test</scope>
    </dependency>
    ...
</dependencies>
```

Configurer le proxy

Dans le répertoire .m2, créer un fichier settings.xml et inclure

```
<settings>

    <proxies>
        <proxy>
            <host>proxy.iut-bm.univ-fcomte.fr</host>
            <port>3128</port>
        </proxy>
    </proxies>

</settings>
```

Serveur de gestion de version

gitLab

- mis à disposition par le CRI pour déposer vos projets
- URL : gitlab.iut-bm.univ-fcomte.fr
- identification LDAP