

Jenkins



Serveur d'intégration continue
multi-langage et multi-plateforme



Grégory BOISSINOT
(@gboissinot)
24/09/2011

A mon propos

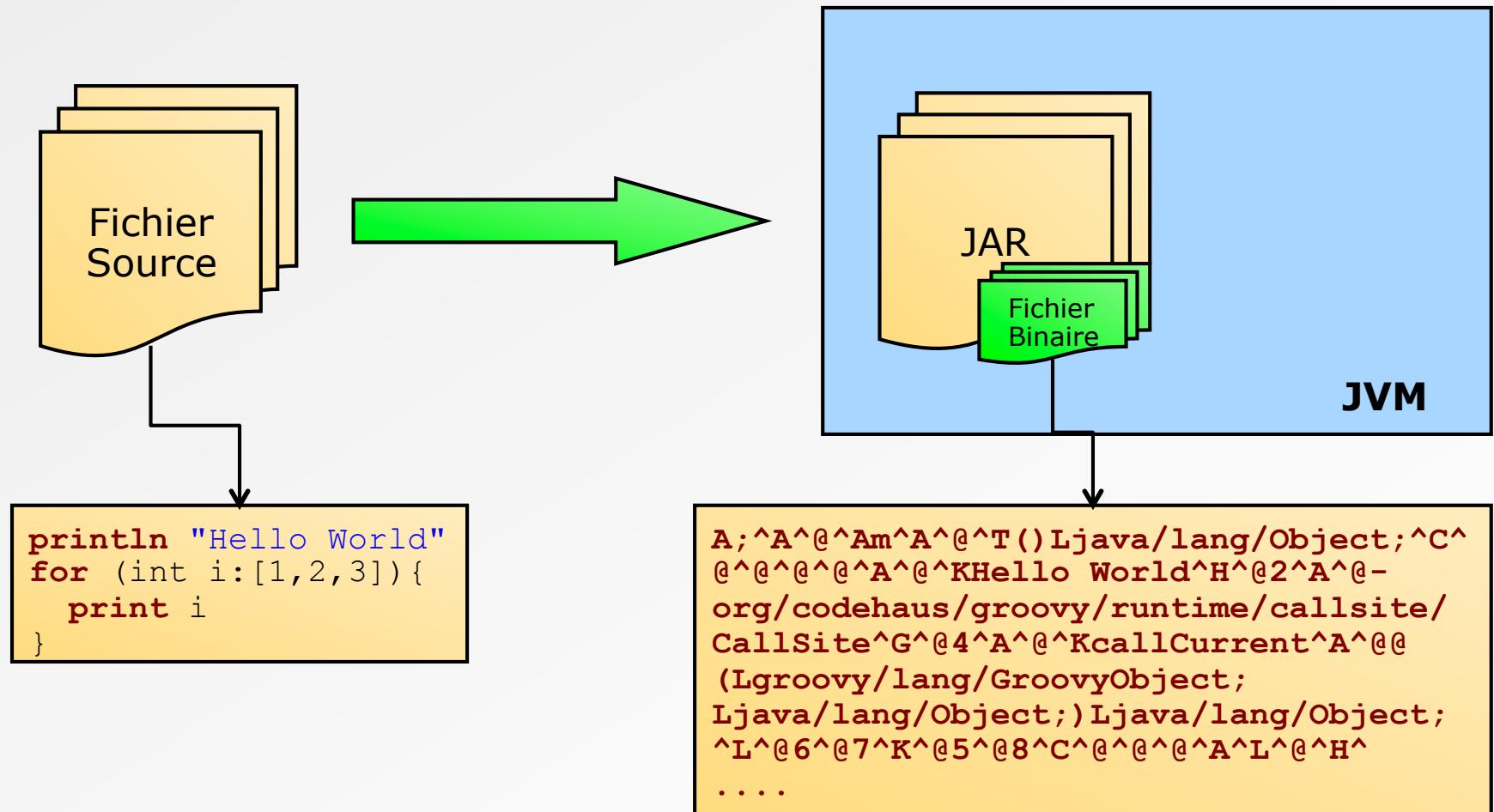
- Consultant et formateur Zenika
 - En charge du comité technique Intégration Continue
 - Intervient chez de nombreux industriels
- Committer Hudson/Jenkins
 - Intégration des chaînes de build Java (Gradle), C/C++, ADA
 - Maintenance corrective et évolutive des plugins réalisés
 - Développement de plugins transverses (xtrigger, envinject, dry-run, ...)
- Développement de différents composants pour la mise en place de processus d'intégration
- Contributeur Gradle

L'intégration continue (IC) en quelques mots

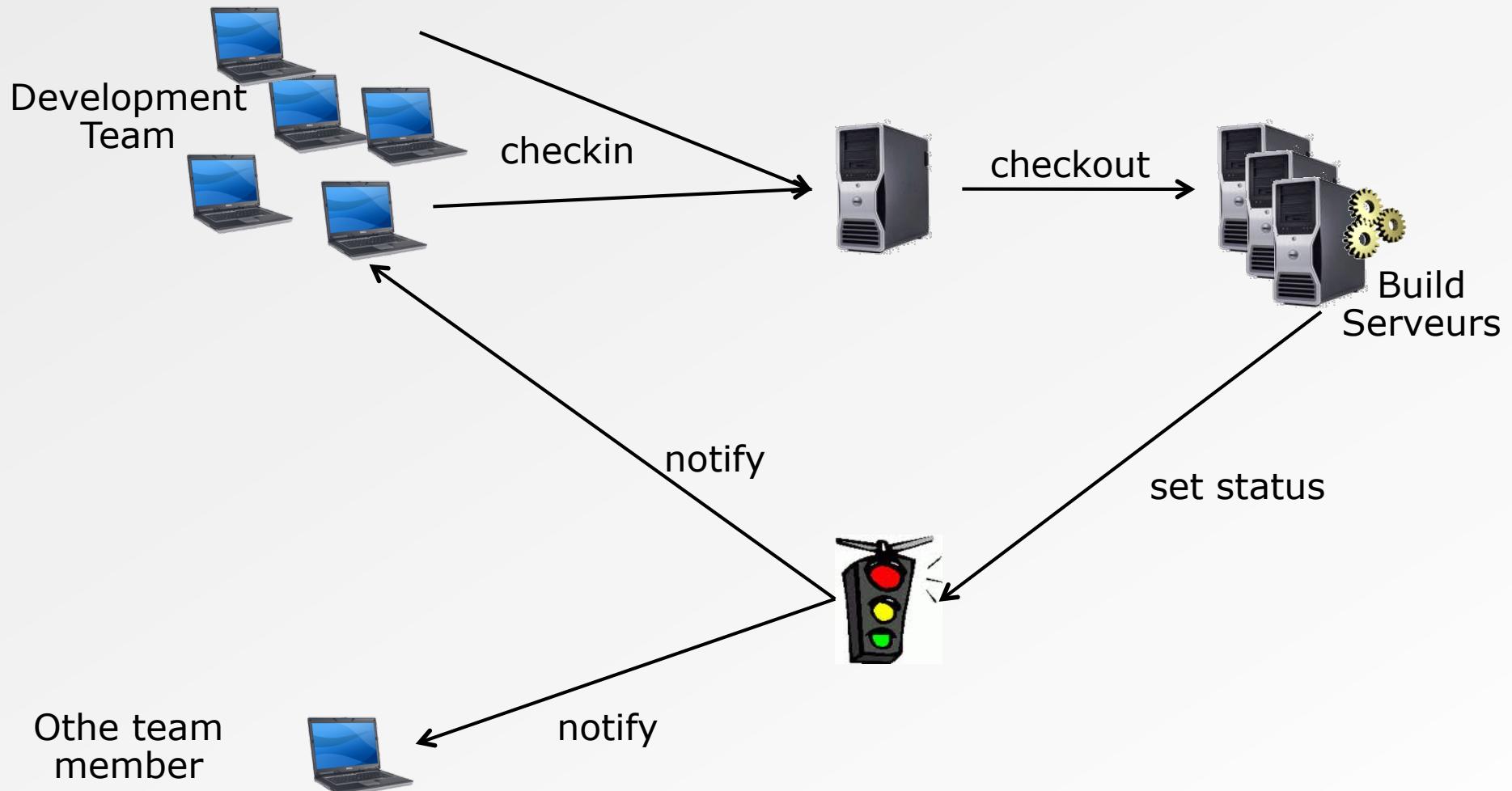
- Pratique de génie logiciel consistant à intégrer le travail des équipes de développement à travers un processus automatisé pour chaque changement d'environnement:
 - Ajout ou modification de fichier dans un gestionnaire de configuration logiciel
 - Ajout ou modification d'un fichier de configuration
 - Ajout ou modification d'un fichier d'infrastructure
- Le travail d'intégration est constitué de différentes étapes: génération de code, compilation, exécution des tests, inspection de code ...
- La fréquence d'intégration réduit un ensemble de risques
- L'objectif est que l'intégration devienne un non événement

Principe du build

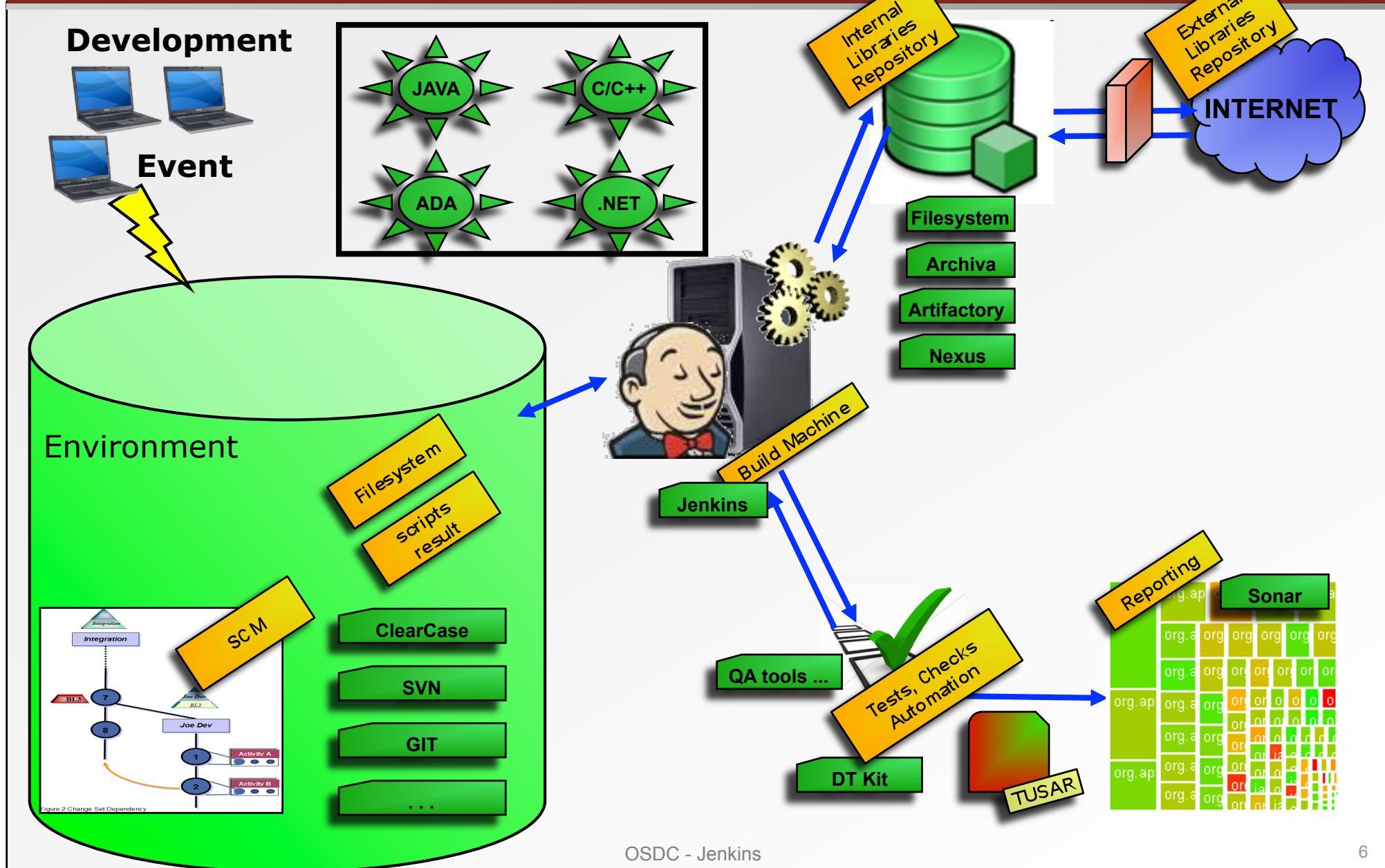
- Transformation d'une production humaine (le code source) en une représentation machine (le binaire)



Première approche de l'IC



Infrastructure d'intégration continue



Jenkins



- Serveur d'intégration continue créée par Kohsuke Kawaguchi
- Ecrit en Java et Groovy
- Très simple à configurer à travers son interface graphique
- Un vaste écosystème de plugins open source en licence MIT
 - > 300 plugins
 - Modèle de contribution très ouvert
- Conçu initialement pour servir des chaînes d'intégration pour des cibles Java, ses plugins lui permettent d'adresser des chaînes d'intégration de projets C/C++, Python, Ada, Ruby, PHP, ... avec le même niveau d'intégration

Historique de Jenkins

2006

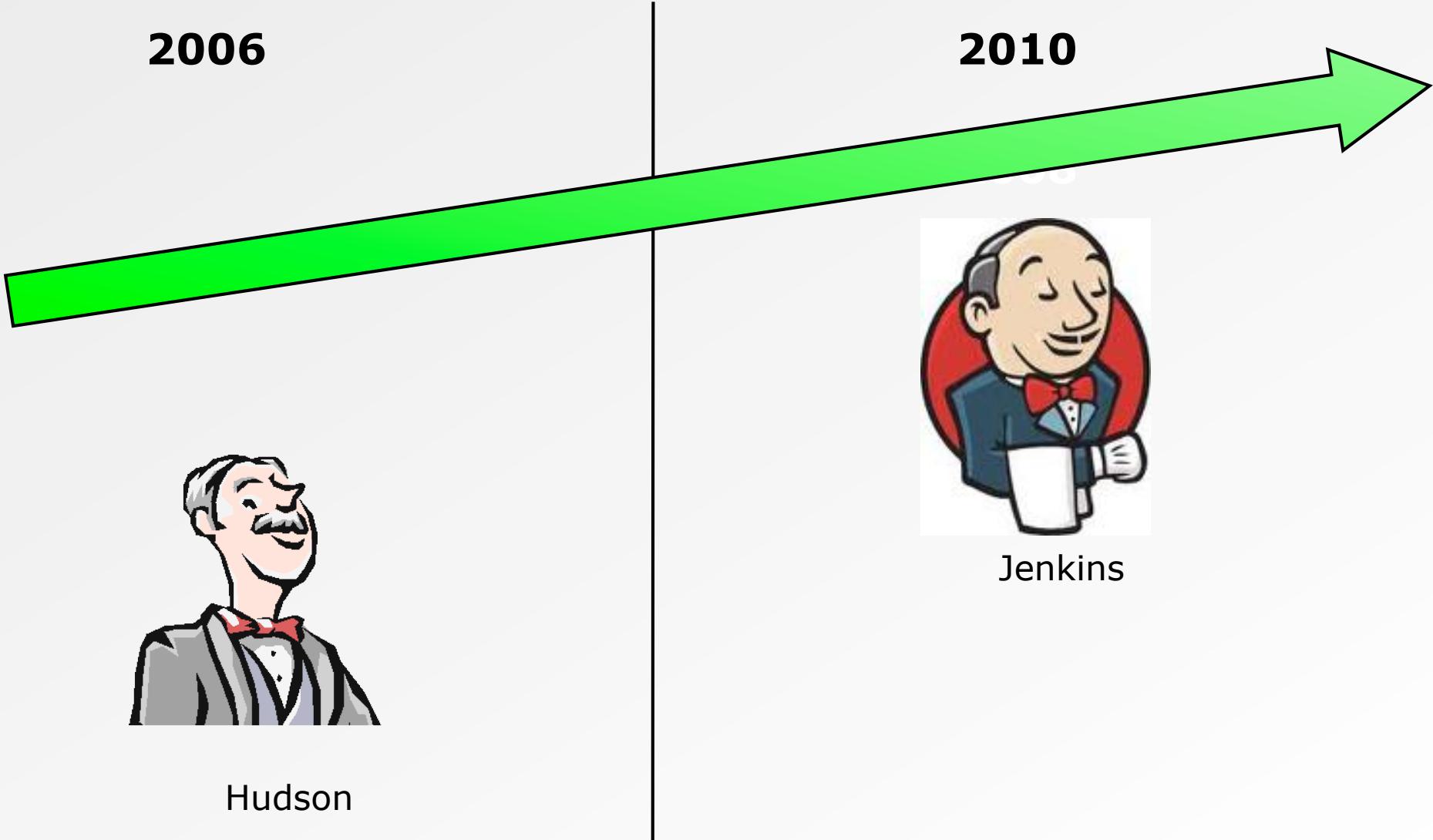


Hudson

2010



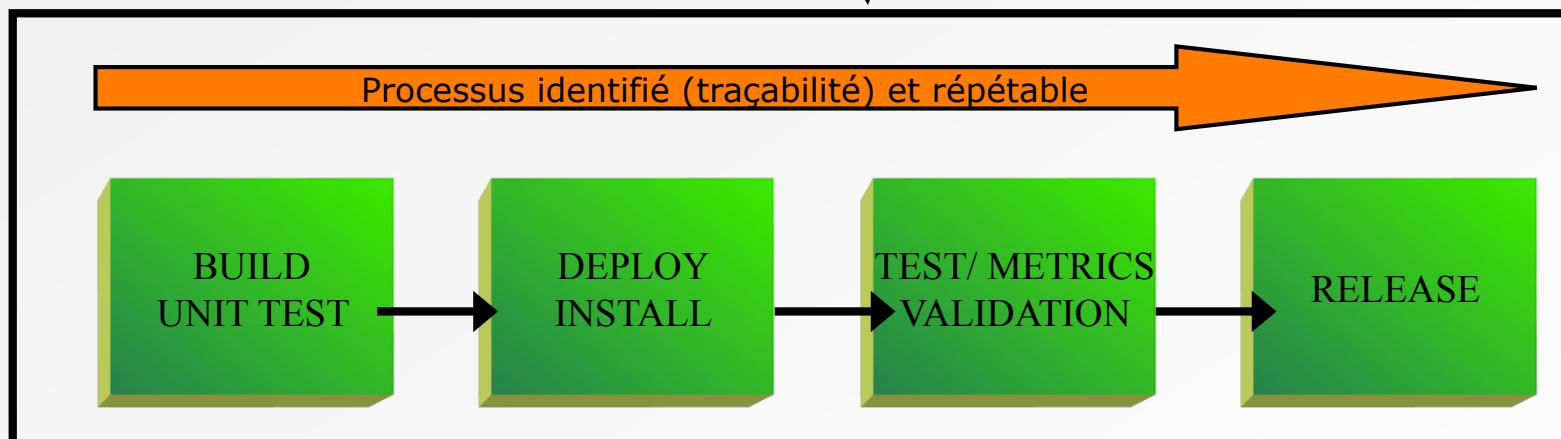
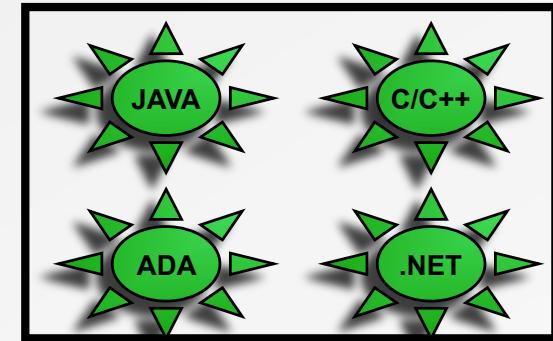
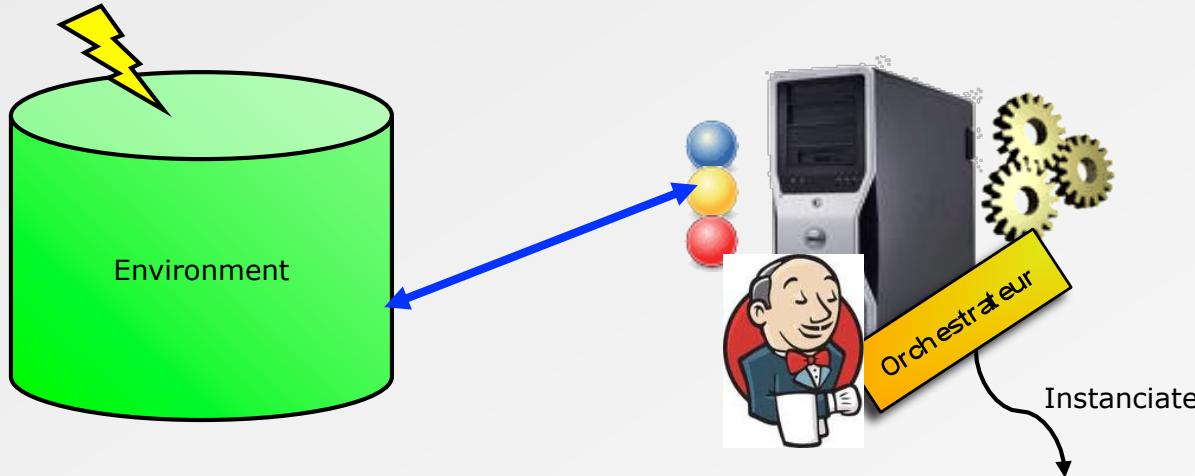
Jenkins



Les plugins Jenkins

- Révolutionne l'extensibilité du noyau Jenkins
 - Mécanisme de points d'extension
- La quasi totalité des plugins sont open-source sous licence MIT
- Très peu de barrières pour contribuer
- Le code source est géré principalement sous GitHub
 - Facilite les contributions à travers des "pull request"

Les ingrédients d'un processus d'intégration



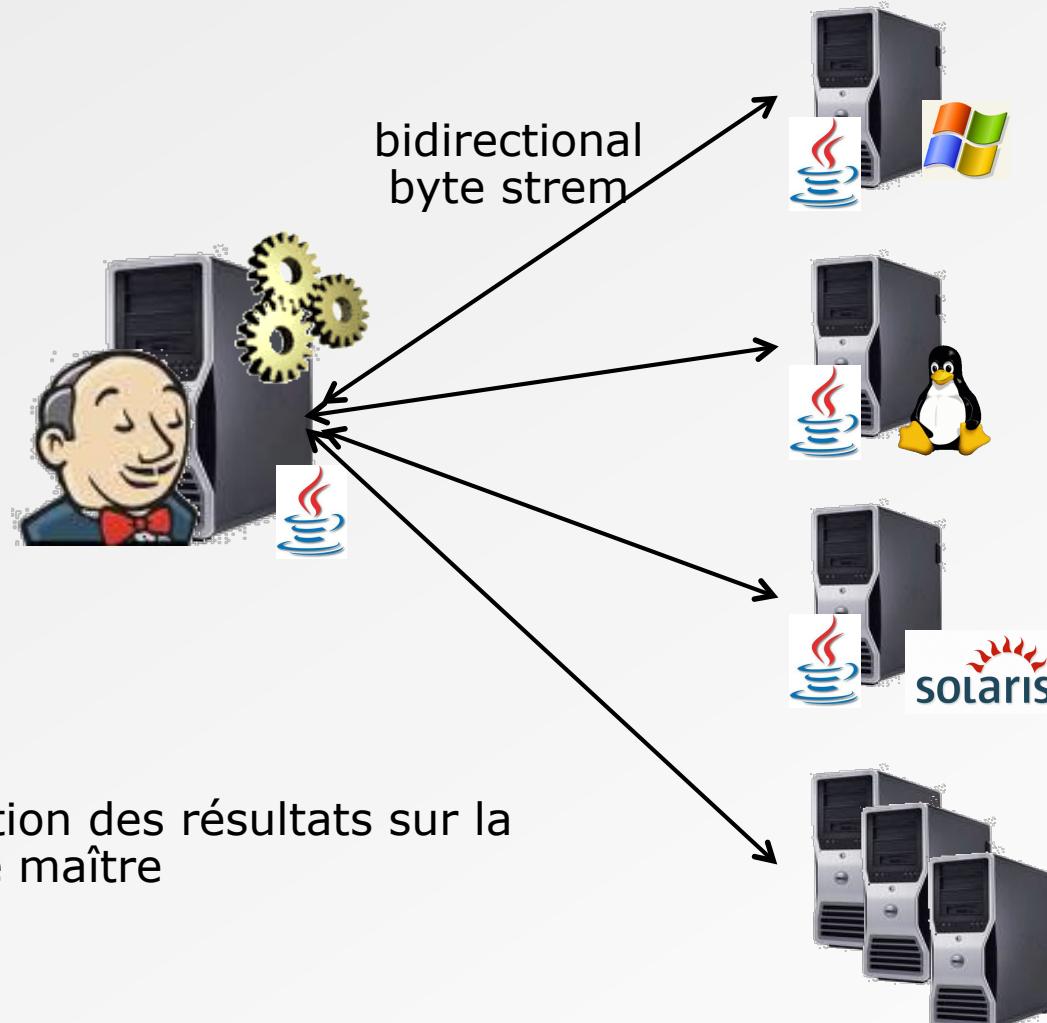
Problématique d'intégration dans un CI serveur

Problématique d'intégration:

- avec votre plateforme
- avec un environnement : SCM, FilleSystem, ...
 - Gestion des différentes typologies et des outils
- avec les différents outils de build
 - Mettre en place un environnement de construction
- avec les outils de métriques
 - Mettre en place un environnement d'exécution
 - Propagation des résultats et visualisation des résultats
 - Pouvoir gérer les différents outils

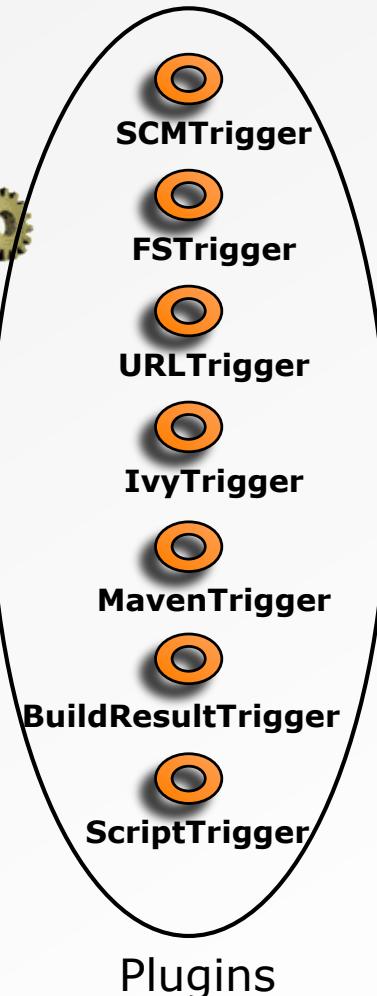
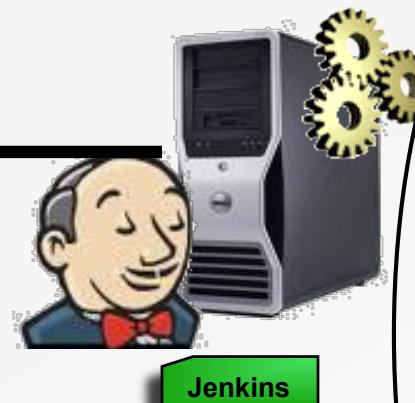
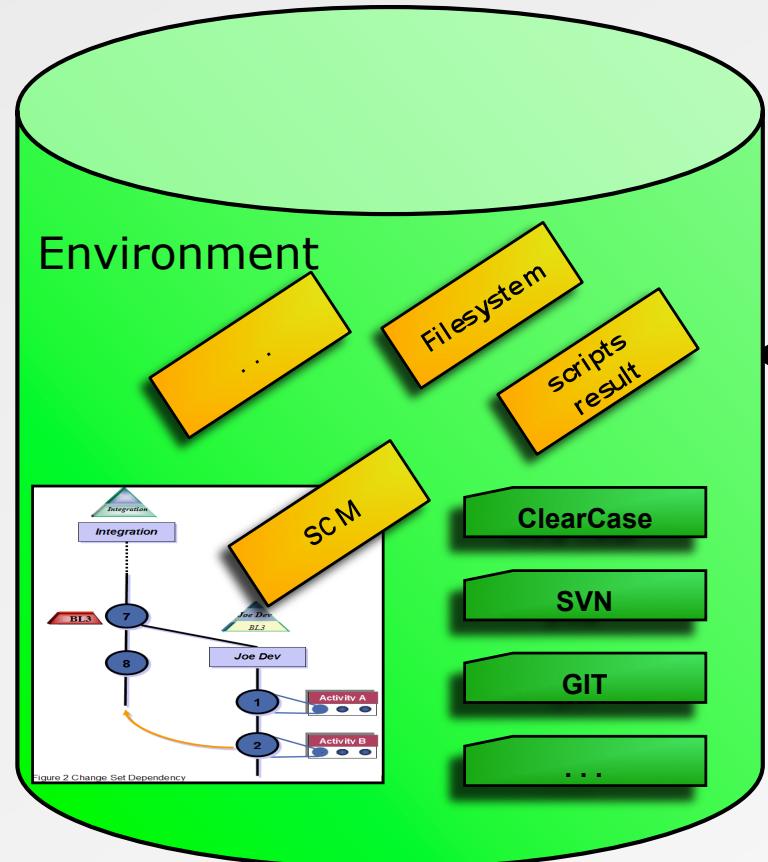
Distribution du build (via master/slaves)

- Notion de machine maître et de n machines esclaves



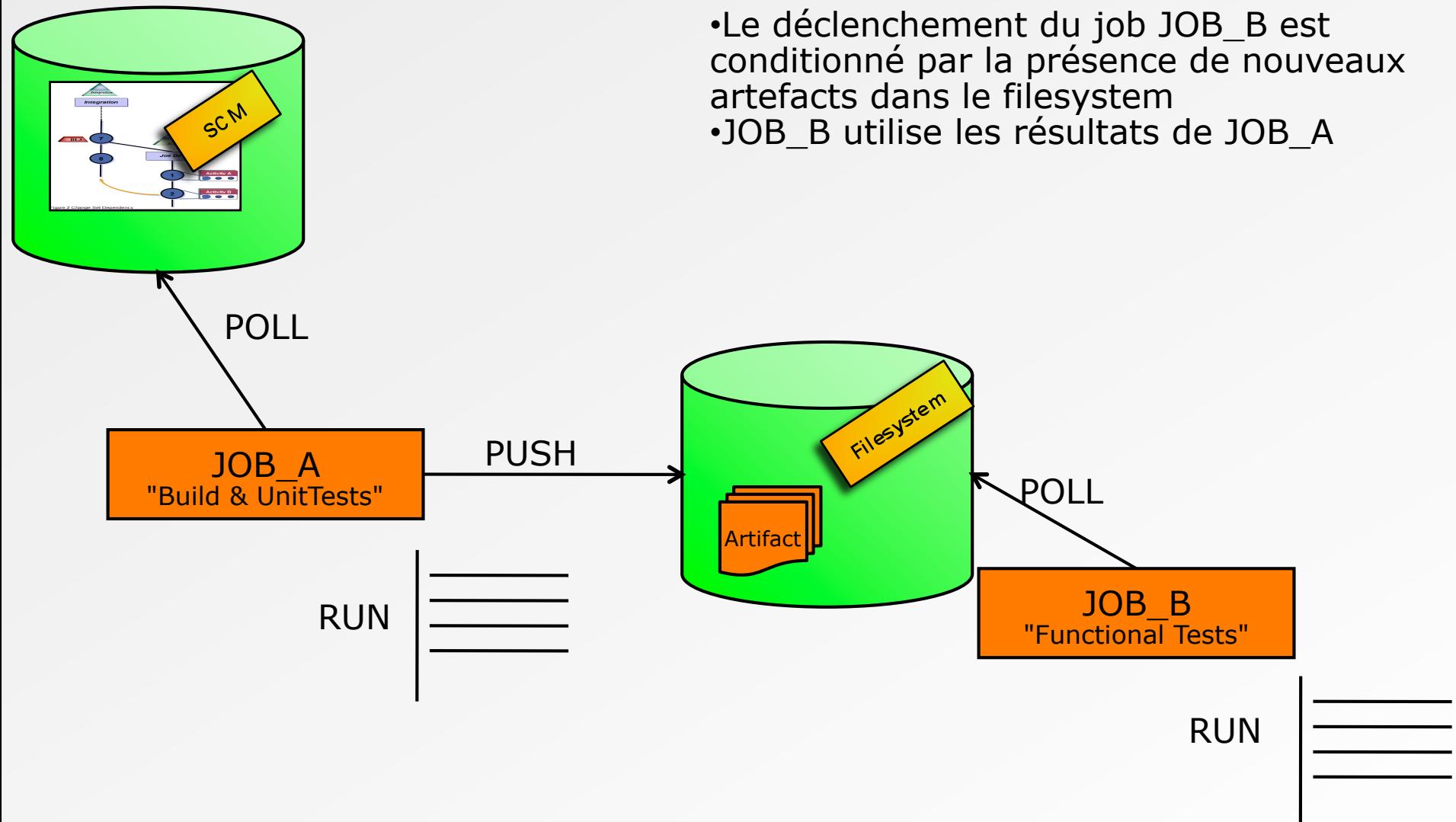
- Agrégation des résultats sur la machine maître

Surveillance des différentes typologies d'environnement

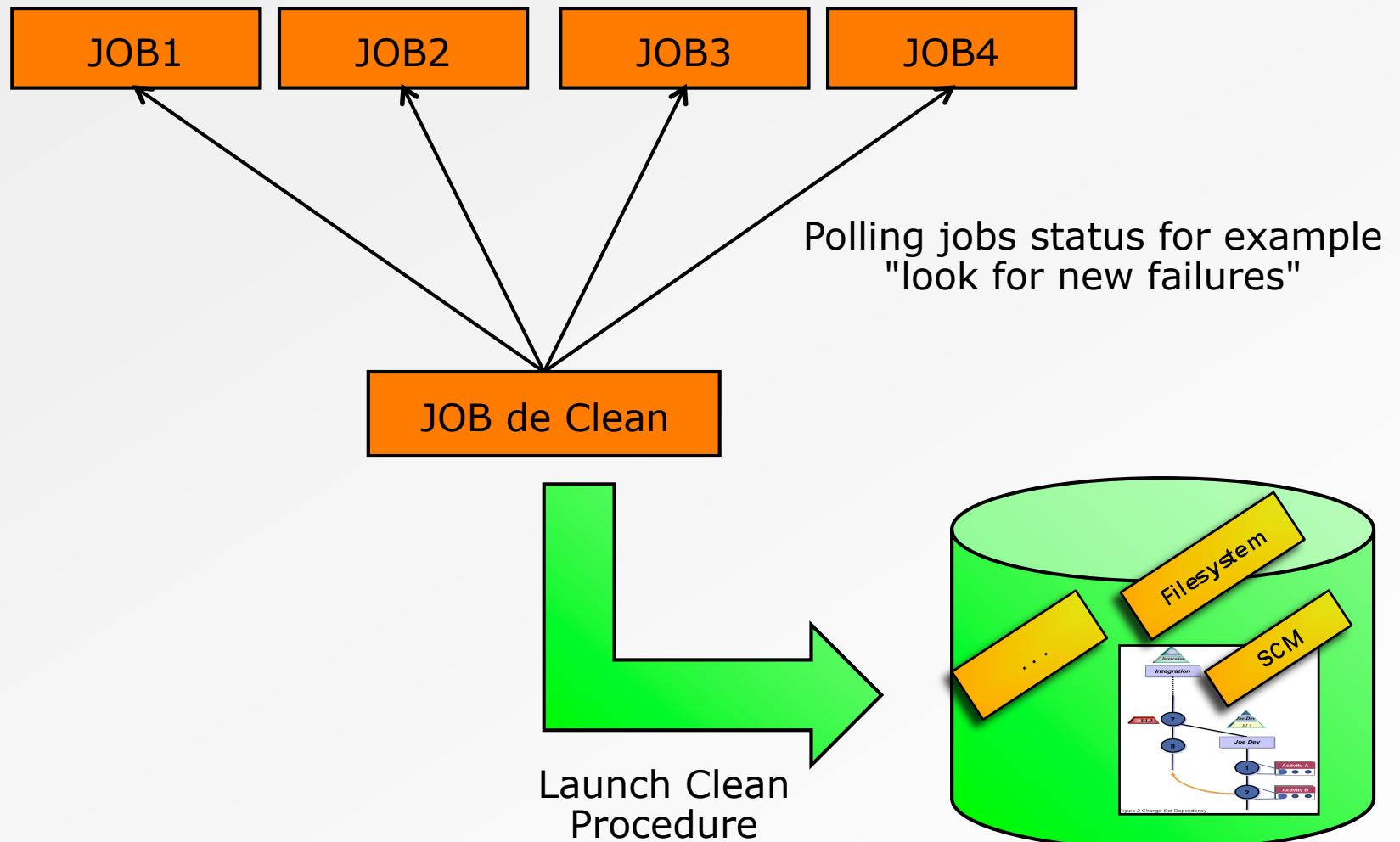


- Au delà de la simple surveillance d'un gestionnaire de version
- Permet de couvrir les différentes conditions de déclenchement d'un job

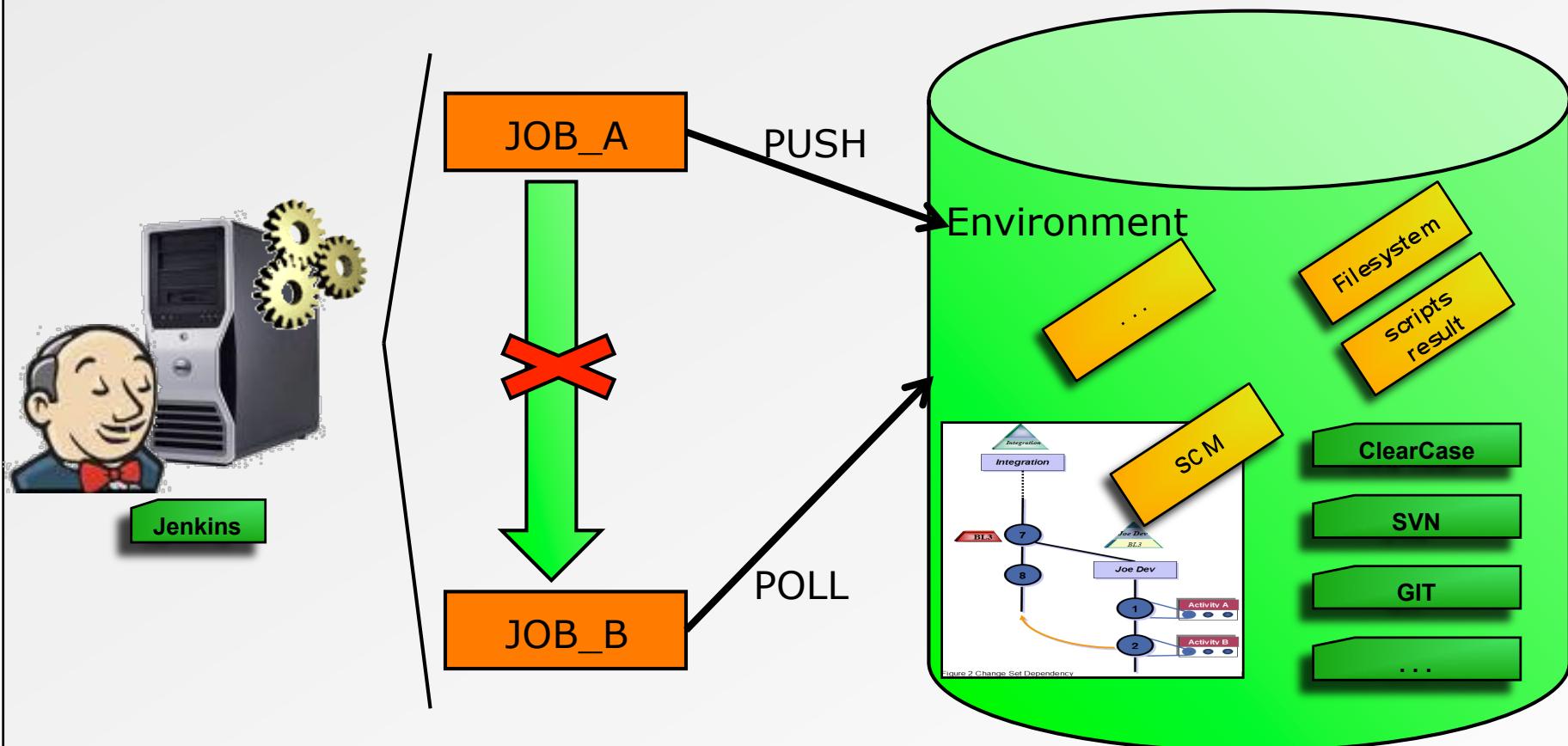
XTrigger – Exemple de Use case – Mise en œuvre d'un "Build Pipeline"



XTrigger – Jobs de remise en état d'un environnement suite à l'échec de job

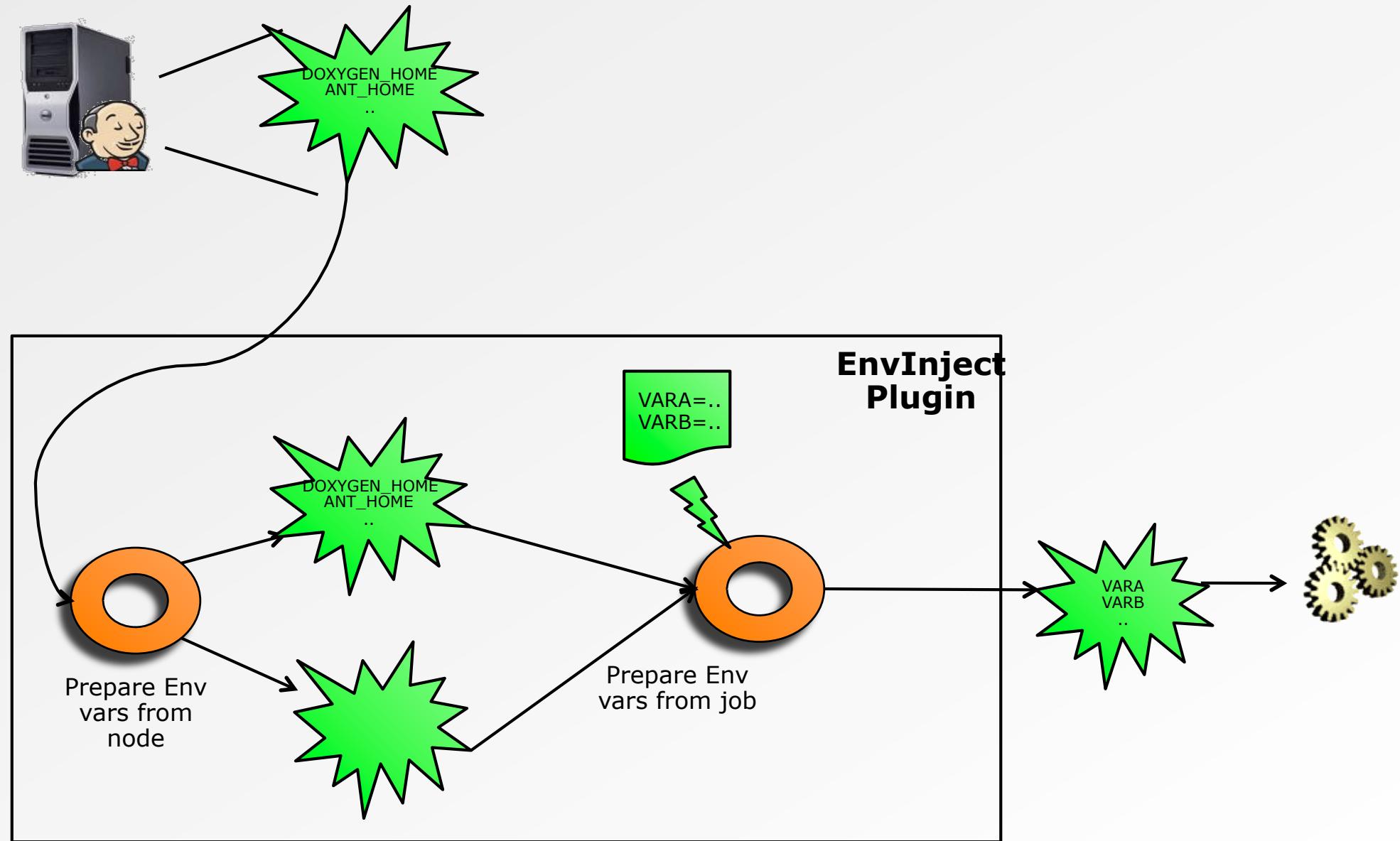


Mise en place des bonnes pratiques d'enchaînement des jobs Jenkins



- Les dépendances statiques entre les jobs sont supprimées
- La dépendance entre les jobs est déléguée aux changements de l'état d'une ressource externe qui est l'entrée de JOB_B
- JOB_B ne se déclenche que si JOB_A s'est exécuté et à modifié la ressource en entrée de JOB_B

Contrôler les variables d'environnement de vos processus IC



Son écosystème de plugins pour le Java



- **Support des projets Java**
 - Builders
 - *Ant, Maven, Gant, Gradle, EasyAnt*
 - Outils de tests
 - *JUnit, TestNG, Selenium*
 - Outils de métriques
 - *Violations (checkstyle, pmd/cpd, findbugs, ...), JavaDocs, Coverage (Cobertura, Clover, Emma)*

Son écosystème de plugins pour le Ruby



- **Support des projets Ruby**
 - Builders
 - *Ruby, Rake*
 - Outils de métriques
 - *Ruby metrics*

Son écosystème de plugins pour le PHP



- **Support des projets PHP**
 - Builder
 - *Phing*
 - Outils de tests
 - *xUnit (PHPUnit)*
 - Outils de métriques
 - *Checkstyle (PHP_CodeSniffer)*
 - *Clover PHP (PHPUnit)*
 - *Dry (phpcpd)*
 - *JDepend (PHP_Depend)*

Plus d'info: <http://jenkins-php.org/>

Son écosystème de plugins pour .NET



- **Support des projets .NET**
 - Builders
 - *MSBuild*
 - Outils de tests
 - *xUnit(MSTest, NUnit, Gallio)*
 - Outils de métriques
 - *Violations (stylecop, fxcop, ...)*

Son écosystème de plugins pour ADA



- **Support des projets ADA**
 - Builders
 - *Gnatmake, Gprbuild*
 - Outils de tests
 - *xUnit (AUnit)*
 - Outils de métriques
 - *Gnatcheck, Gnatmetric*

Son écosystème de plugins pour le C/C++



- **Support des projets C/C++**
 - Builders
 - *CMake, SCons*
 - Outils de tests
 - *xUnit (CppUnit, BoostTest, UnitTest++, ...)*
 - Outils de métriques
 - *CCCC, Doxygen, Cppcheck, C++Test, Klocwork et DTKit*

Exemple de mise en œuvre avec Cppcheck

Publish Cppcheck results

Cppcheck report XMLs

Cppcheck must be configured to generate XML reports for this plugin to function. [Fileset includes](#) setting that specifies the generated cppcheck XML report files, such as */cppcheck-result-*.xml. Basedir of the fileset is relative to the module root unless you have configured your SCM with multiple modules, in which case it is relative to the workspace root. Note that the module root is SCM-specific, and may not be the same as the workspace root. If no value is set, then the default */cppcheck-result.xml is used. Be sure not to include any non-report files into this pattern.

Build status

			Total		New		Total		New
Thresholds:	<input type="text" value="1"/>								

Configure the build status and health. A build is considered as unstable or failure if the new or total number of errors exceeds the specified thresholds. The build health is also determined by thresholds. If the actual number of errors is between the provided thresholds, then the build health is interpolated.

severity.evaluation

Severity 'error'
 Severity 'possible error'
 Severity 'style'
 Severity 'possible style'

Determines which severity of errors should be considered when evaluating the build status and health.

Graph configuration

Chart Width Chart Height

Display all errors
 Display Severity 'error'
 Display Severity 'possible error'
 Display Severity 'style'
 Display Severity 'possible style'

Determines which severity of errors should be displayed in the graph.

Build #21 (Aug 30, 2009 8:12:49 PM)

No changes.

Started by user [anonymous](#)

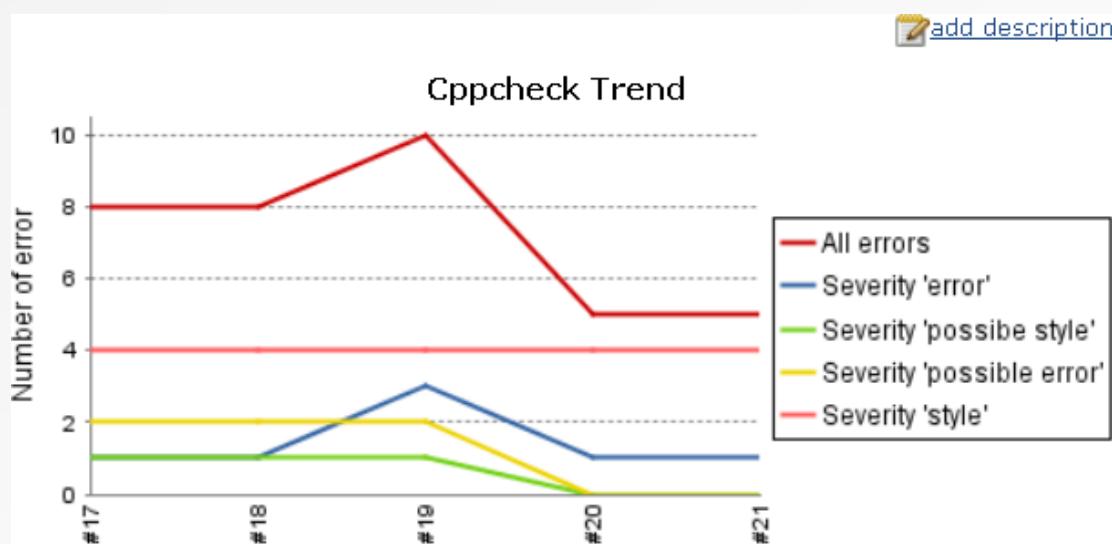
Cppcheck: [5 errors](#).

- No new error

Permalinks

- [Build number](#)

[add description](#)



Solution d'intégration pour Python



- **Support des projets Python**

- Environnement d'exécution

- A mettre en étape de build

- *virtualenv pour créer un environnement contrôlé avec un set de 'site-package'*
 - *pip pour la gestion et l'installation de cet environnement contrôlé avec le envInject pour la mise en place*

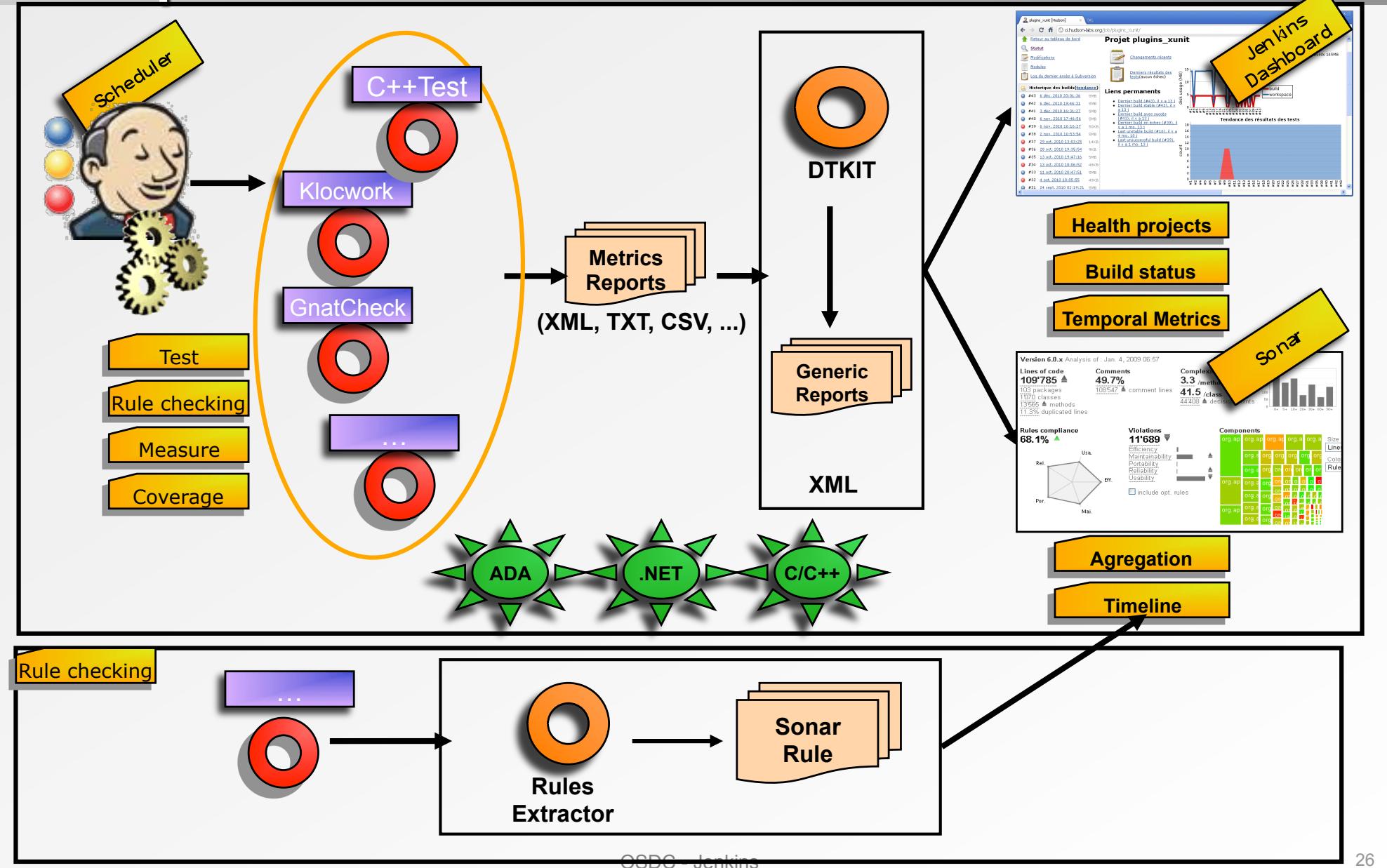
- Exécution de scripts Python

- *Python et Jython*

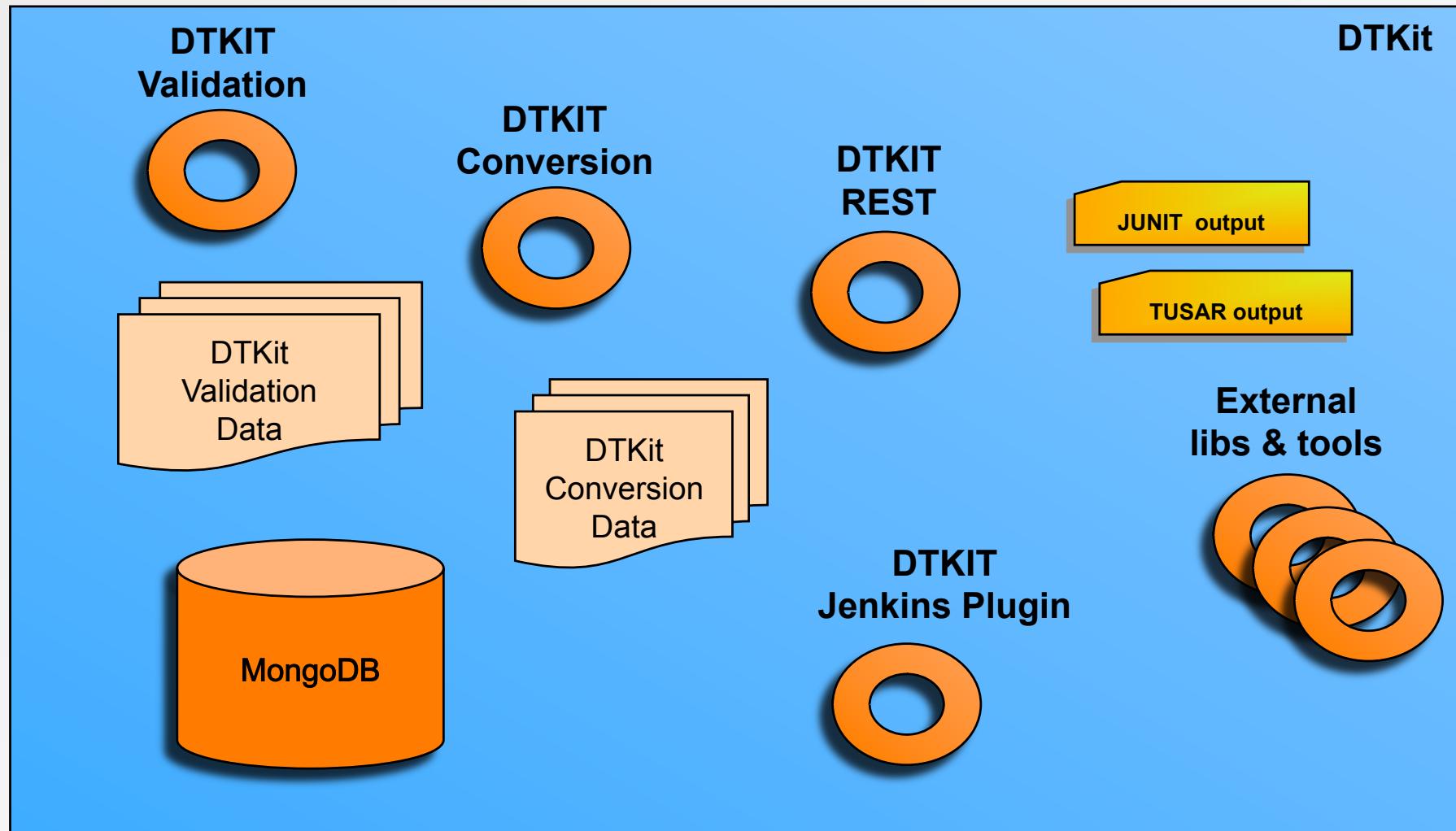
- Outils de métriques

- *Violations (Pylint, Clone digger, pep8)*
 - *Sloccount*

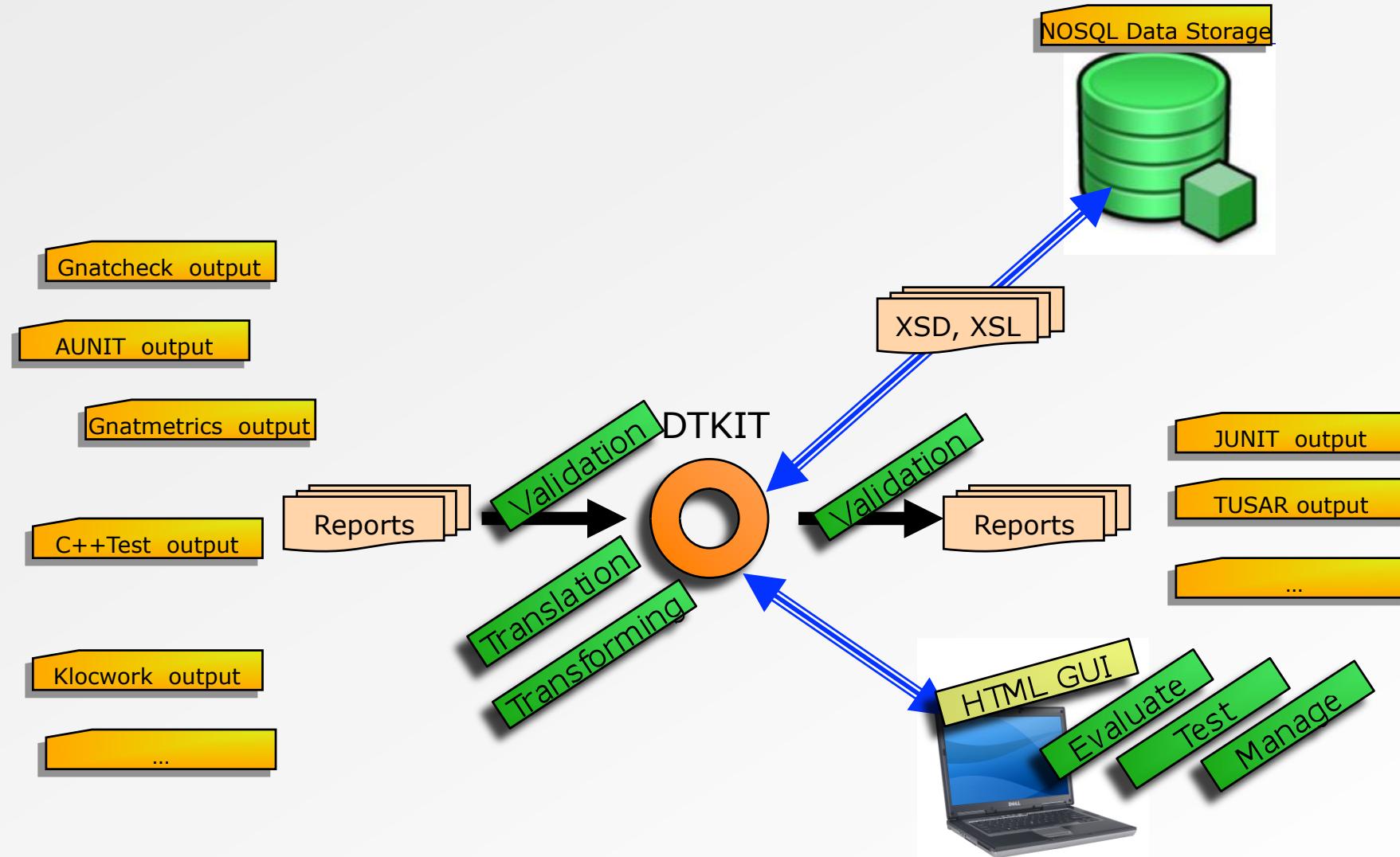
Solution uniforme d'intégration des métriques



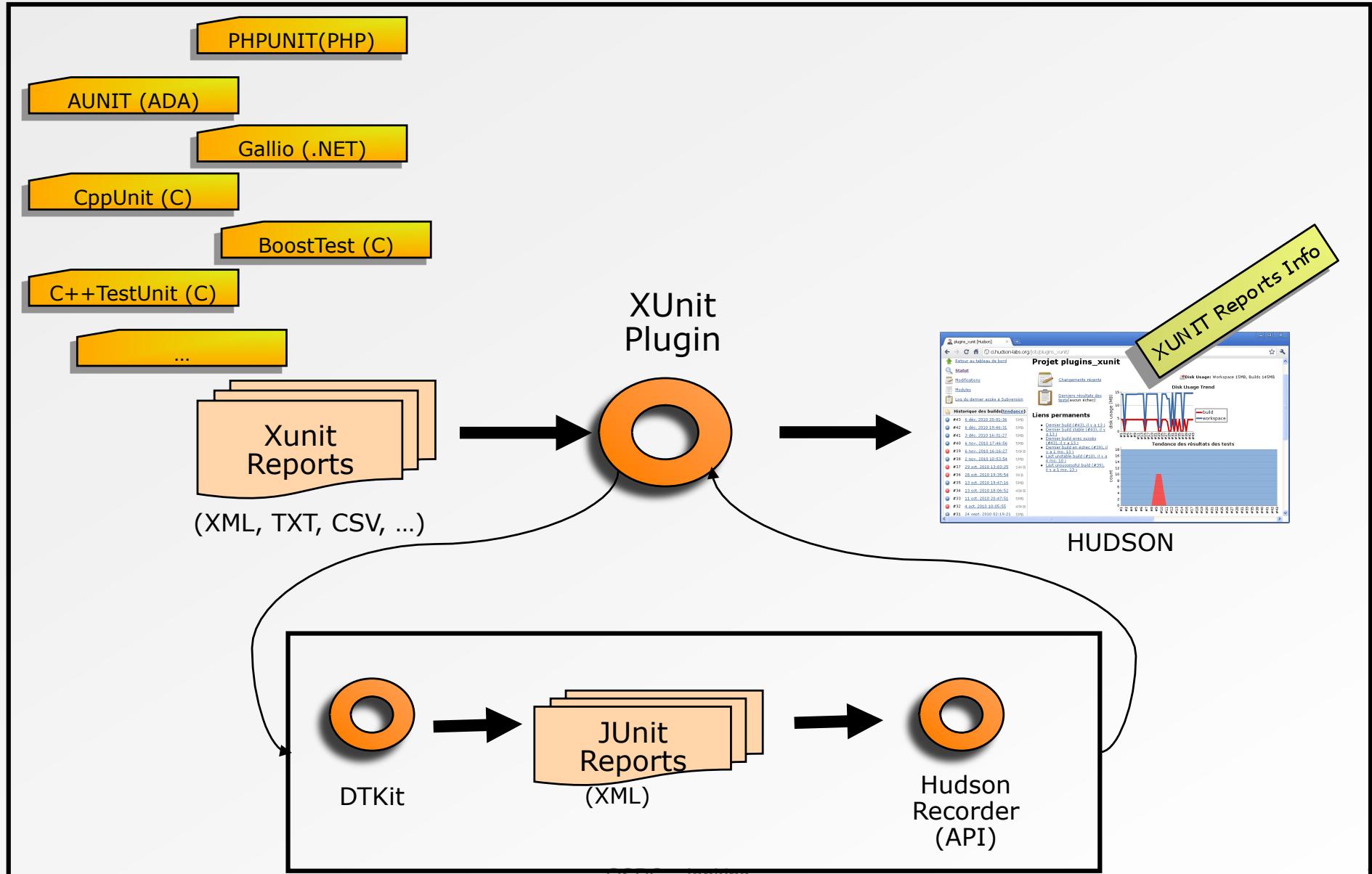
DTKit – Une brique de traitement uniforme de qualité de code



DTKit – Workflow



Le framework d'intégration xUnit



Plateforme Android

- Génération et utilisation d'un émulateur pour la durée du build
- Se combine parfaitement avec les Matrix Project
- Plus d'information sur le Jenkins Emulator Plugin
<https://wiki.jenkins-ci.org/display/JENKINS/Android+Emulator+Plugin>



Jenkins & le Cloud

- Plusieurs plugins d'intégration



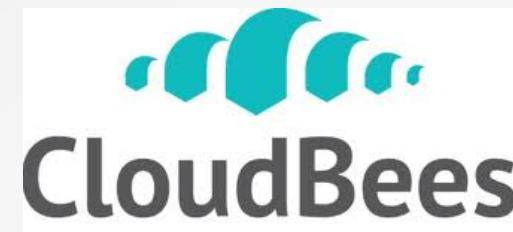
A screenshot of the Jenkins AWS plugin interface. At the top, there's a navigation bar with tabs: "Credentials" (highlighted in blue), "Images" (highlighted in red), "Store", and "Users". Below the navigation bar is a table listing EC2 images:

ID	Name
emi-CDF01003	image/debian.5.0.x86-64.img.manifest.xml
eri-8044134E	initrd/initrd.img-2.6.28-11-generic.manifest.xml
ekl-45A9126A	kernel/vmlinuz-2.6.28-11-generic.manifest.xml

Plugin Amazon
EC2



Plugin vmware



Plugin CloudBees
Deployer sur RUN@Cloud



PluginJClouds

Mise à disposition des binaires



- Gestion des différents protocoles

- File

- ArtifactDeployer*

- SSH

- Publish over SSH*

- FTP

- Publish over FTP*

- CIFS

- Publish over CIFS*

Au delà du run : le multi langage pour le développement



- Possibilité de développer un plugin Jenkins en langage Ruby
- Début du support en langage Python
- Le support d'autres langages à terme est possible

Toutes les contributions sont les bienvenues !!

