

L'intégration continue

*Faire de l'intégration
un non-événement*



Jean-Baptiste Defard

NET/PSYS
ingénierie informatique

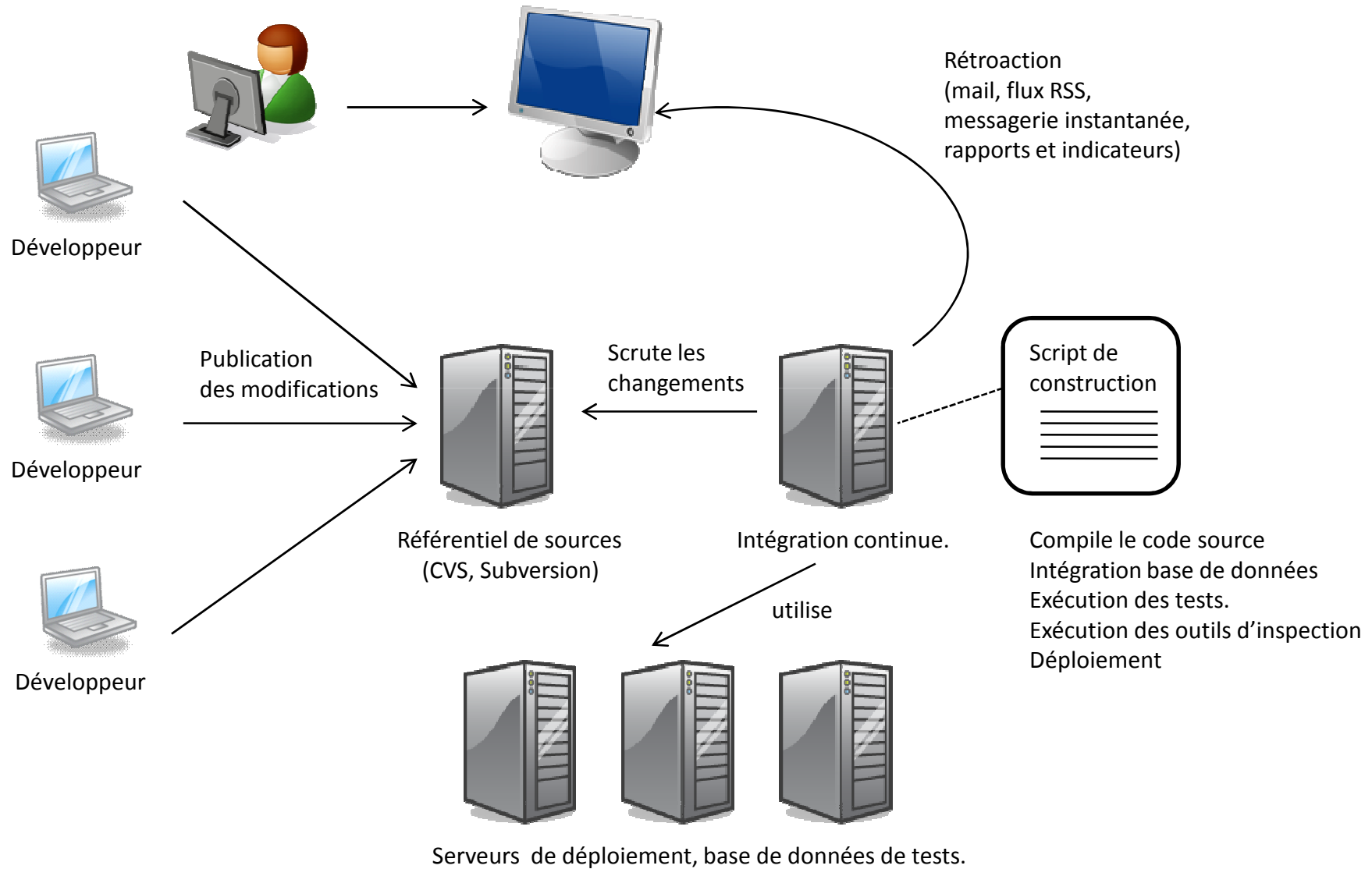
Intégration continue : principe fondamental

Tout changement du code d'un logiciel est immédiatement pris en compte par une construction automatique d'intégration

- ❑ Pratique du développement logiciel
 - ▶ Les membres d'une équipe de développement intègrent leur travail quotidiennement

- ❑ Détection précoce des erreurs
 - ▶ Chaque intégration est vérifiée lors d'une construction automatique, incluant les tests (fonctionnels et techniques)

Infrastructure



Machine de développement



Développeur

- ❑ Les projets sont constructibles avec des **scripts automatisables** (Ant, Maven, Make...)
- ❑ La construction d'un projet est indépendante de l'environnement de développement (IDE)
- ❑ Le développeur exécute une construction privée avant de publier ses changements dans le référentiel de sources

Référentiel de sources



Référentiel de sources
(CVS, Subversion)

- ❑ Définit l'état précis des artefacts du logiciel
- ❑ Interlocuteur unique du robot d'intégration
- ❑ Il contient l'ensemble des artefacts nécessaires à la construction du projet :
 - ▶ Codes sources
 - ▶ Fichiers de configuration
 - ▶ Données particulières (base de données)

Machine d'intégration



Intégration

- ❑ L'automate, exécute la construction d'intégration plusieurs fois par jour, au plus à chaque mise à jour du référentiel de sources
- ❑ 100% des tests existants sont exécutés à chaque construction
- ❑ L'objectif de la construction d'intégration est de produire un **logiciel exécutable** qui peut être déployé et testé fonctionnellement

Infrastructure de support à l'intégration



Serveurs de déploiement, base de données de tests.

- ❑ La construction d'intégration est configurée pour utiliser des bases de données de test conformes à l'environnement cible de production

- ❑ Au besoin, la construction déploie automatiquement le logiciel intégré sur un serveur d'application :
 - ▶ Tests fonctionnels
 - ▶ Prototypes exécutables par les parties prenantes

Rétroactions



- ❑ Les résultats de l'intégration sont notifiés à l'équipe de développement et publiés sur un site Web accessible aux parties prenantes du projet

- ❑ La correction des constructions en erreur est une **priorité** des membres de l'équipe de développement :
 - ▶ Gain de temps
 - ▶ Diminution des coûts

Corriger les défauts détectés avant qu'ils ne se manifestent

Inspection



Intégration continue.

- ❑ L'intégration continue permet d'obtenir automatiquement des indicateurs d'avancement et d'état qualitatif d'un projet en cours de développement

- ❑ Les outils d'assurance qualité participent au contrôle des risques :
 - ▶ Risque de **faible qualité logiciel**
 - ▶ Risque de **découverte tardive des défauts**

Logiciels d'intégration continue.

- ❑ L'offre logicielle est importante.
 - ▶ CruiseControl, AntHill, Continuum, **Hudson** ...

<http://confluence.public.thoughtworks.org/display/CC/CI+Feature+Matrix>

- ❑ Le choix d'Hudson <https://hudson.dev.java.net/> :
 - ▶ Open source sous licence MIT
 - ▶ Projet mature avec une communauté active (java.net)
 - ▶ Facile à installer et à utiliser
 - ▶ Large choix de plugins
 - ▶ Constructions distribuées et suivi des versions (multi-module)
- ❑ Un tutoriel en français
<http://linsolas.developpez.com/articles/hudson/>



DEMONSTRATION

Outils d'assurance qualité

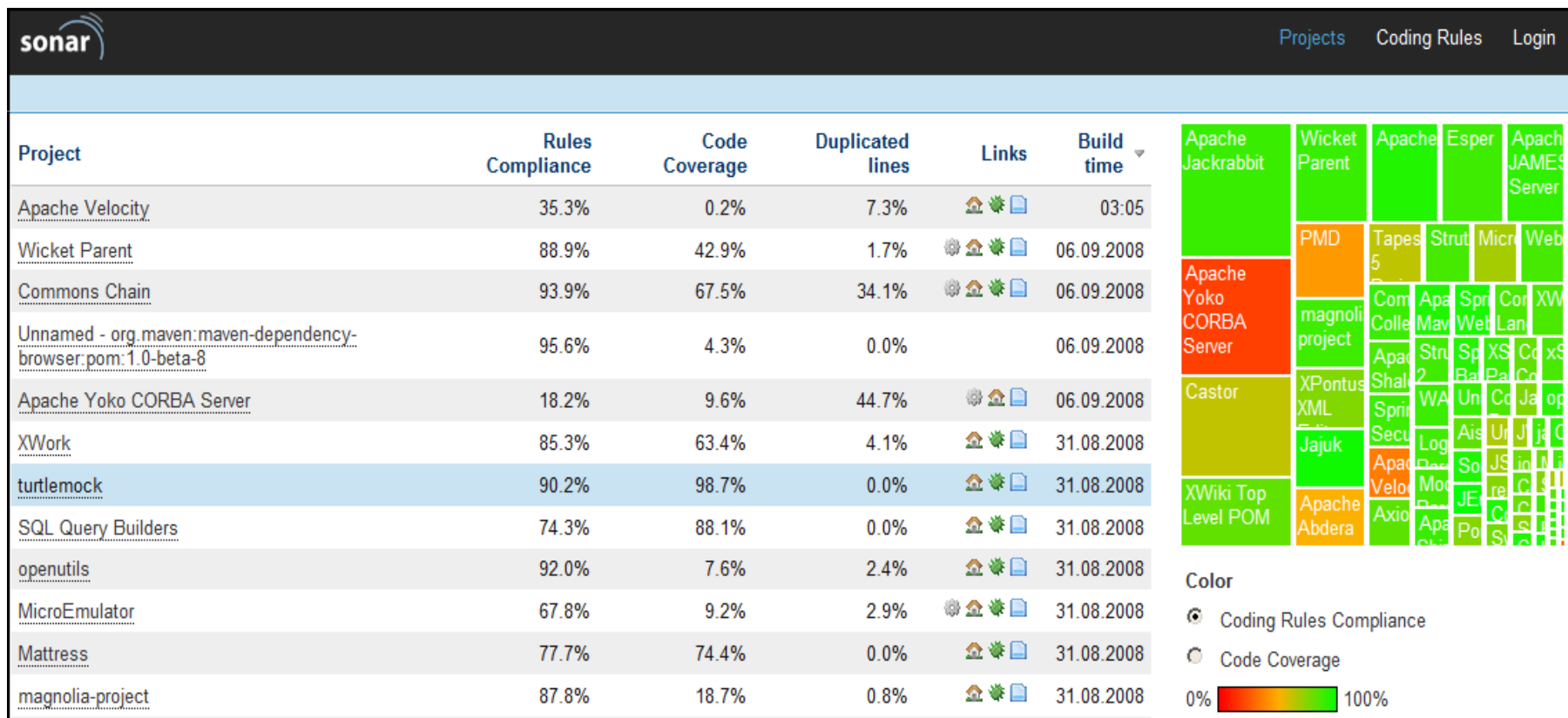
L'exécution régulière des tests et des outils d'inspection permettent de recueillir les informations pour l'évaluation de la complexité, la détection des codes potentiellement « malicieux »,...

Checkstyle PMD	Respect des règles de codage et des bonnes pratiques
CPD Simian	Contrôle l'absence de copier/coller
JDepend.	Analyse de dépendance et respect des règles d'architecture
JNCSS.	Contrôle de la complexité cyclomatique et de la documentation technique

Sonar : la qualité sous contrôle

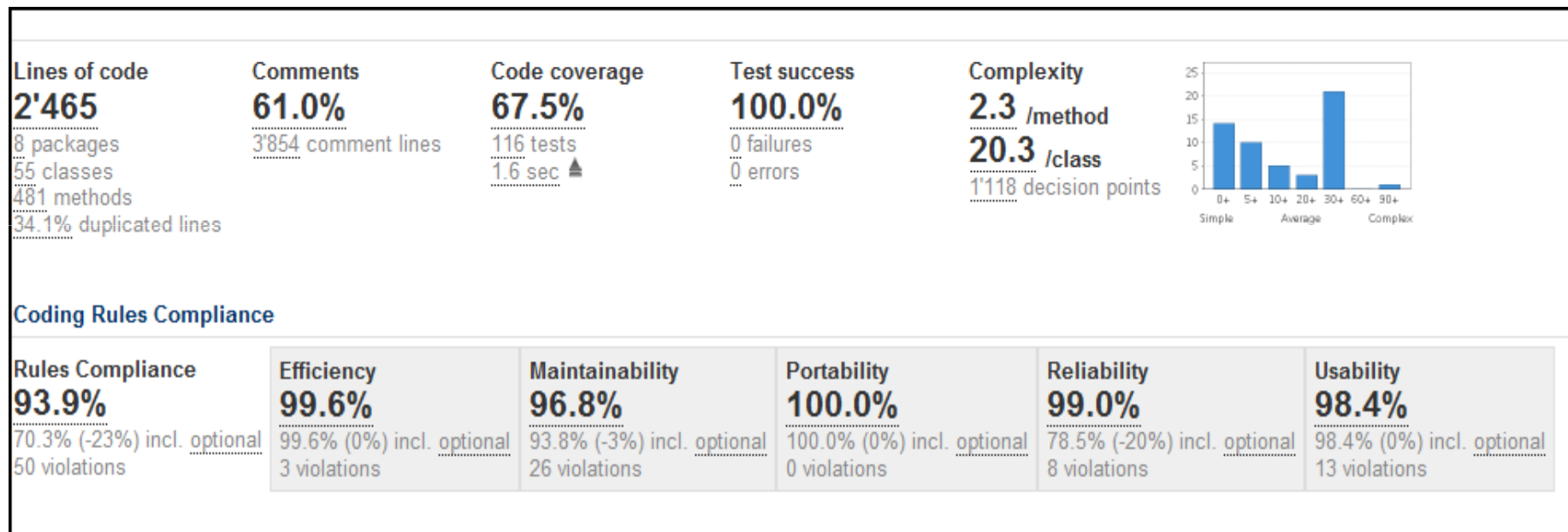
- ❑ Sonar <http://sonar.codehaus.org> est un projet open source, disponible depuis le début de cette année. Il s'intègre naturellement dans une construction Maven et dans un environnement d'intégration continue. Il participe à la **rétroaction**.

<http://nemo.sonar.codehaus.org/>



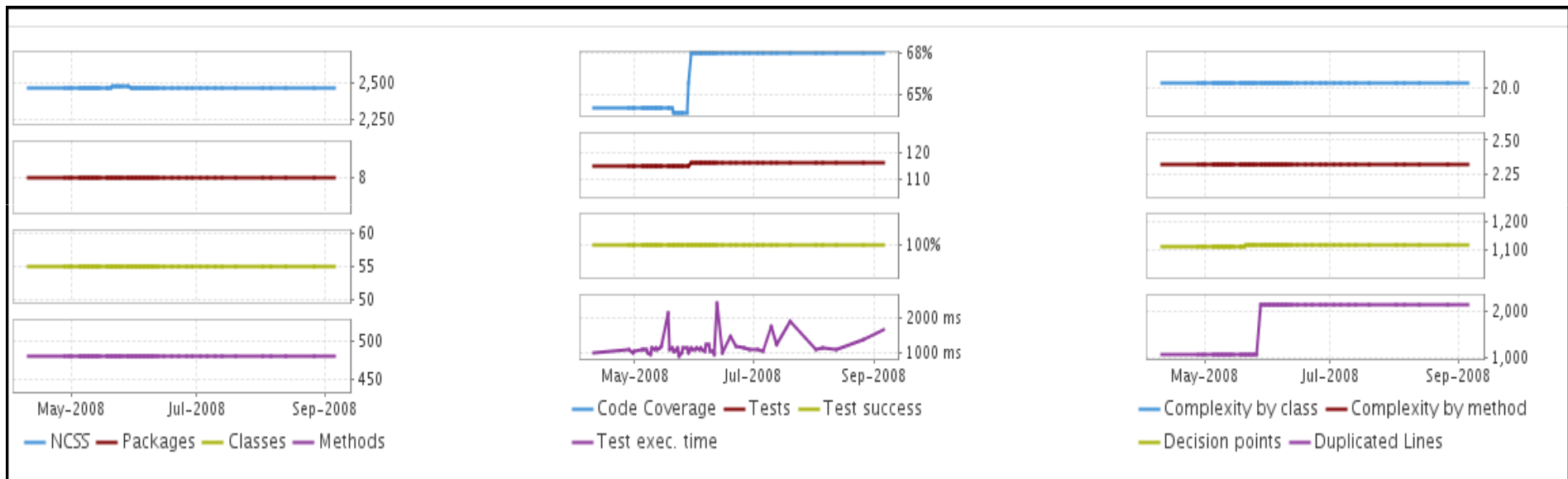
Indicateurs d'assurance qualité.

- ❑ Sonar synthétise les indicateurs d'assurance qualité produits par la construction Maven.



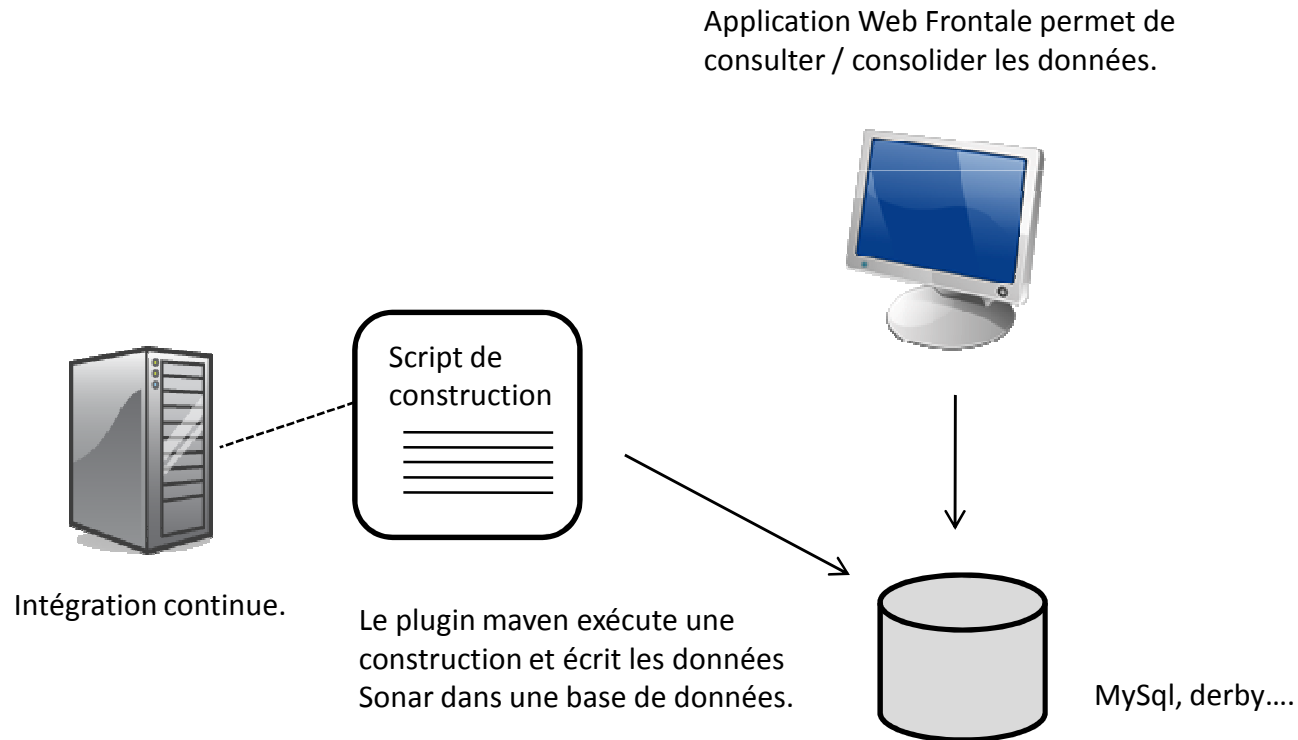
Vision chronologique

- ❑ Sonar fournit une vision **chronologique** de l'évolution du logiciel.



Principes de fonctionnement et architecture.

- ❑ Par conception Sonar s'intègre naturellement dans une infrastructure d'intégration continue.



Sonar : le suivi automatisé



Intégration continue.

- ❑ La collecte des données Sonar est réalisée par un **plugin Maven** lors d'une construction d'intégration d'un projet.
- ❑ Les données sont accessibles au travers d'une application Web, qui permet la consultation et la **comparaison** des indicateurs.

Sonar est un véritable portail d'accès aux données d'intégration.



DEMONSTRATION

Conclusion

- ❑ L'intégration continue est une pratique agile proposée pour améliorer la qualité et la productivité des développements.
- ❑ Après plus d'un an d'utilisation, Hudson est une « bonne » solution, simple à mettre en œuvre, facile à faire évoluer et qui répond à un large éventail de besoins.
- ❑ Associé à Sonar, cette plateforme fournit une solution intéressante pour le pilotage et le suivi des développements

QUESTIONS ?



Je vous remercie de votre attention.

BUFFET



