NETAPSYS
ingénierie informatique

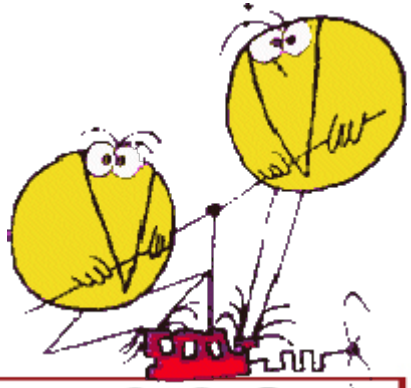
L'Intégration Continue & Agilité…

" …des outils efficaces. "

Agile NANTES
-
Mars 2010

Qui sommes nous ?

- Fabian PIAU – fabian.piau@netapsys.fr
  - Ingénieur développement chez NETAPSYS
  - Diplômé MIAGE Nantes 2009
- Sébastien ROUL – sebastien.roul@netapsys.fr
  - Directeur NETAPSYS Atlantique
  - 15 ans de développement
  - Co-fondateur : www.nantes-jug.org



Sommaire

- Problématique
- Agilité et Intégration Continue
  - Les concepts
  - Fonctionnement général
  - Les outils utilisés chez Netapsys
- Retour d'expérience
  - Vision Développeur
  - Vision Chef de projet
- Conclusion

Livrer un logiciel

- De qualité
  - Avec le moins de bogues possibles (encore moins des régressions…)
  - Complètement testé (idéalement…)
  - En assurant une maintenance facile à long terme
- Dans un temps restreint
  - Le plus rapidement
  - Mise en commun du développement → Plusieurs personnes → Besoin d'intégrer
  - Réutiliser des composants existants → Développement modulaire
- Avec un suivi rigoureux
  - Transparence pour le client
  - Rétroaction

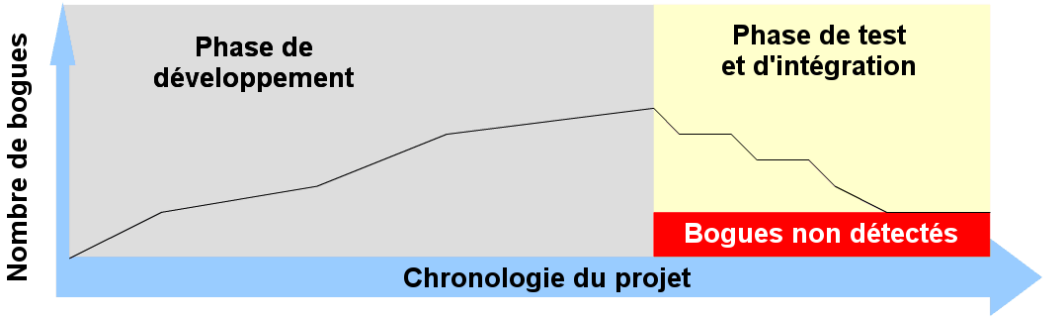
Un outil : L'Agilité

- Des principes
  - Accepter le changement et l'accueillir favorablement
  - Apprendre à apprendre
  - Viser l'excellence technique
  - Remettre en cause les pratiques régulièrement
- Des valeurs
  - La communication et la transparence
  - Le courage et la responsabilité
  - La solidarité et l'esprit d'équipe
  - L'humilité
  - La rétroaction
  - La simplicité

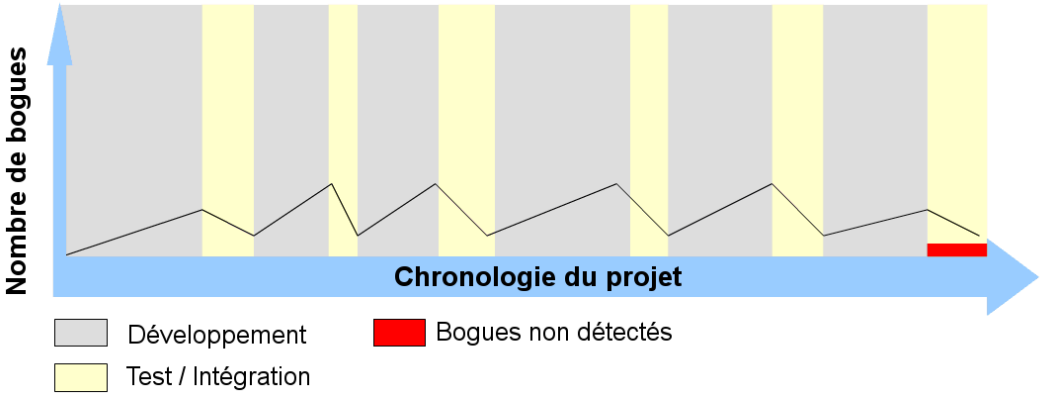
Un autre outil : l'intégration Continue (IC)

- Une méthode agile ☺
- Principe: plus l'intégration est fréquente, moins elle est longue

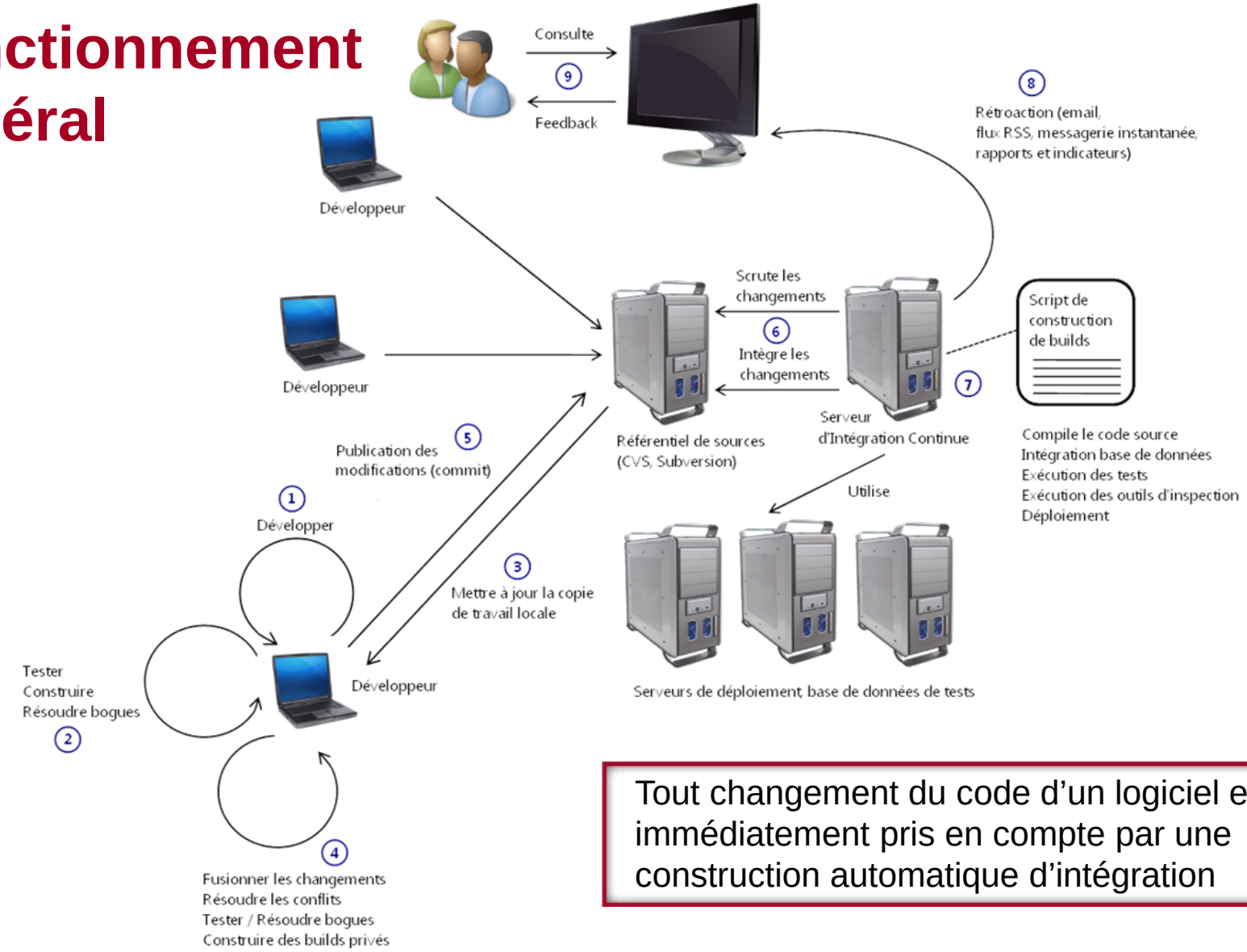
- Sans IC



- Avec IC



Fonctionnement général



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

Construire des builds tous les jours…

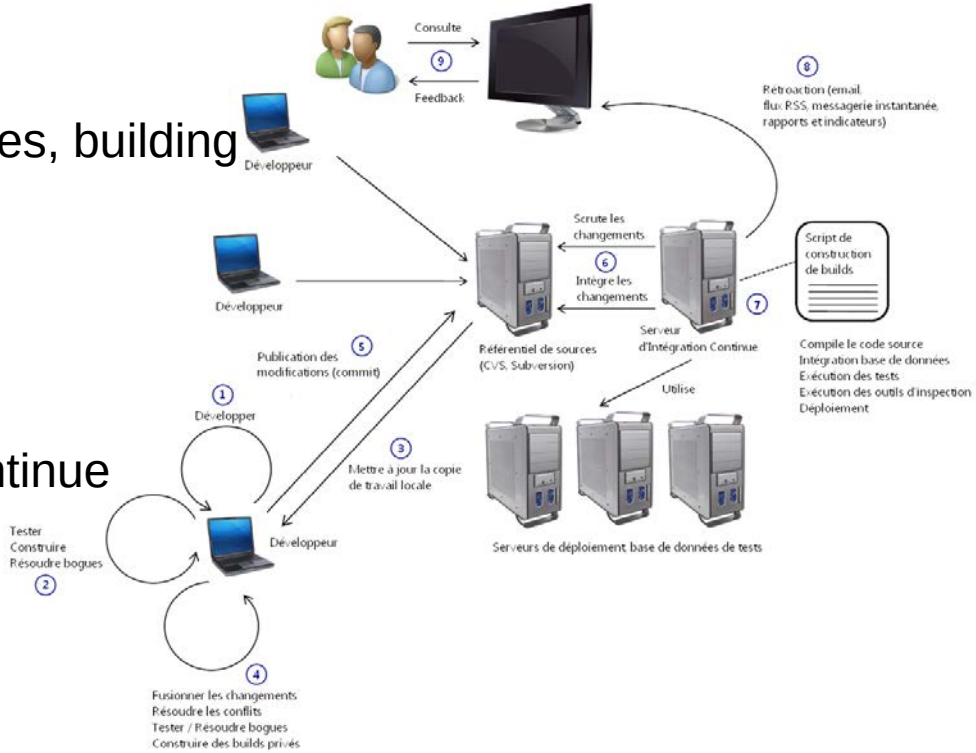
- Chargement de la dernière version du projet depuis le gestionnaire de version
- Compilation
- Exécution des tests unitaires
- Inspection du code (en vue de générer les métriques de qualité)
  Génération de la documentation, des rapports, des notes de release (par exemple la Javadoc, rapport Checkstyle)
- Construction de la release
- Déploiement de l'application sur l'environnement d'intégration
- Exécution des tests d'intégration

…Mais des builds différents!

- Build local / privé
  - J'exécute un build sur ma machine
    - Compilation
    - Tests unitaires
- Build d'intégration
  - Le serveur d'intégration exécute un build
    - Idem build local
- Build de nuit
  - Le serveur exécute un build complet
    - + Tests d'intégration
    - + Documentation, rapports, métriques
    - + Release / Déploiement

Les outils utilisés

- Un gestionnaire de versions
  - SVN
- Gestionnaire de dépendances, building
  - Maven
- Tests unitaires
  - jUnit, Mockito, Eclemma
- Un serveur d'intégration continue
  - Hudson
- Repository manager
  - Artifactory
- Outils de qualimétrie
  - Sonar



Les outils utilisés

- Hudson, la « météo » de l'intégration continue
  - Un onglet = un projet
  - Différents types de build par projet

Hudson

search | fabian.piau|Déconnexion

Hudson » CRA

ANNULER LE RAFRAICHISSEMENT AUTOMATIQUE

ajouter une description

- Nouvelle tâche
- Administrer Hudson
- Personnes
- Historique des builds
- Éditer la vue
- Supprimer la vue
- Relations entre les projets
- Vérifier les empreintes numériques
- My Views
- Leader board

All | CRA | +

S	W	Tâche ↓	Dernier succès	Dernier échec	Dernière durée
		cra	2 mo. 9 j (#275)	N/A	13 mn
		cra-principal	2 mo. 9 j (#278)	N/A	48 s
		cra-site	2 mo. 9 j (#69)	N/A	16 mn
		cra-sonar	2 mo. 8 j (#167)	N/A	3 mn 19 s
		cra-support	2 mo. 27 j (#29)	N/A	9,7 s

Icône: S M L

Légende | tous les builds | tous les échecs | pour les derniers builds seulement

File d'attente des builds

Pas de build en attente.

État du lanceur de build

#	État
1	En attente
2	En attente

Page générée le :1 mars 2010 15:10:21 Hudson ver. 1.346

Le plugin Chuck…

Hudson search S'identifier

Hudson » LCPC-Build ACTIVER LE RAFRAICHISSEMENT AUTOMATIQUE

Retour au tableau de bord
Statut
Modifications
Espace de travail

Historique des builds(tendance)
#89 16 mars 2010 08:59:39
#88 15 mars 2010 20:09:27
#87 10 mars 2010 19:57:35
#86 10 mars 2010 19:42:24
#85 5 mars 2010 16:03:00
tous les builds tous les échecs

Chuck Norris doesn't use a computer because a computer does everything slower than Chuck Norris.

Espace de travail
Changements récents

Projets en aval

LCPC-Deploy

Liens permanents

- Dernier build (#89), il y a 1 j 5 h
- Dernier build stable (#89), il y a 1 j 5 h
- Dernier build avec succès (#89), il y a 1 j 5 h

Les outils utilisés

- Configuration d'un build
  - Scrutage du SVN

#267 2 déc. 2009 11:39:12
#266 1 déc. 2009 11:48:23
tous les builds tous les échecs

Avancé...

Gestion de code source

Aucune
Subversion
Modules
URL du repository
Répertoire local du module (optionnel) cra-core
Ajoutez d"autres emplacements...

Utiliser 'svn update'
Quand cette option est activée, Hudson utilisera 'svn update' à chaque fois que cela est possible, ce qui rend le build plus rapide. Attention, les artefacts du build précédent seront conservés au démarrage d'un nouveau build.

Revert
If checked, Hudson will do 'svn revert' before doing 'svn update'. This slows it down, but will prevent files being modified from build to build.

Navigateur de la base de code (Auto)
Avancé...

CVS

Ce qui déclenche le build

Construire à la suite d'autres projets (projets en amont)
Noms des projets cra-principal
Plusieurs projets peuvent être spécifiés en les séparant par des virgules: abc, def
Déclencher les builds à distance (Par exemple, à partir de scripts)
Construire périodiquement
Scruter l'outil de gestion de version
Planning */5 * * * *

Les outils utilisés

- Configuration d'un build
  - La commande Maven à lancer, les notifications par email, etc.

Build

Invoquer les cibles Maven de haut niveau

Version de Maven: maven-2.0.9

Cibles Maven: -Pinteg clean package tomcat:undeploy tomcat:deploy-only

Avancé... Supprimer

Ajouter une étape au build

Actions à la suite du build

- [ ] Publier les Javadocs
- [ ] Archiver des artefacts
- [ ] Consolider les résultats des tests en aval
- [x] Publier le rapport des résultats des tests JUnit

XML des rapports de test: **/target/surefire-reports/*.xml

Une configuration du type Fileset 'includes' qui indique où se trouvent les fichiers XML des rapports de test, par exemple 'myproject/target/test-reports/*.xml'. Le répertoire de base (basedir) du fileset est la racine du workspace.

- [ ] Construire d'autres projets (projets en aval)
- [ ] Enregistrer les empreintes numériques des fichiers pour en suivre l'utilisation
- [ ] Publish Testability Explorer Report
- [ ] Update relevant JIRA issues
- [x] Notifier par email

Destinataires: [illegible]

Liste des destinataires, séparés par un espace. Un email sera envoyé lors d'un échec d'un build.

- [x] Envoyer un email pour chaque build instable
- [ ] Envoyer des emails séparés aux personnes qui ont cassé le build

- [ ] Perform Subversion tagging on successful build
- [x] Continuous Integration game

Les outils utilisés

- Build de nuit
  - Utilisé pour la génération du Maven site

cra-core

Dernière publication: 12/21/2009 cra-core

Documentation sur le projet
- Info Projet
- **Rapports Projet**
  - Checkstyle
  - Cobertura Test Coverage
  - CPD Report
  - FindBugs Report
  - JavaDocs
  - JavaNCSS Report
  - JDepend
  - Liste des tags
  - Maven Surefire Report
  - Rapport PMD
  - Simian report
  - Sources Xref
  - Sources Xref des Tests
  - Test JavaDocs

built by: maven

Rapport générés par Maven

Ce document fournit une vue d'ensemble des divers rapports générés automatiquement par Maven Chaque rapport est décrit brièvement ci-dessous.

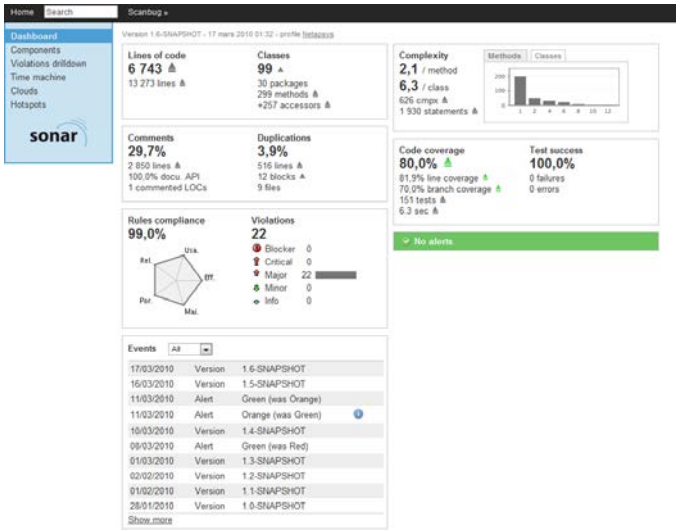
Vue d'ensemble

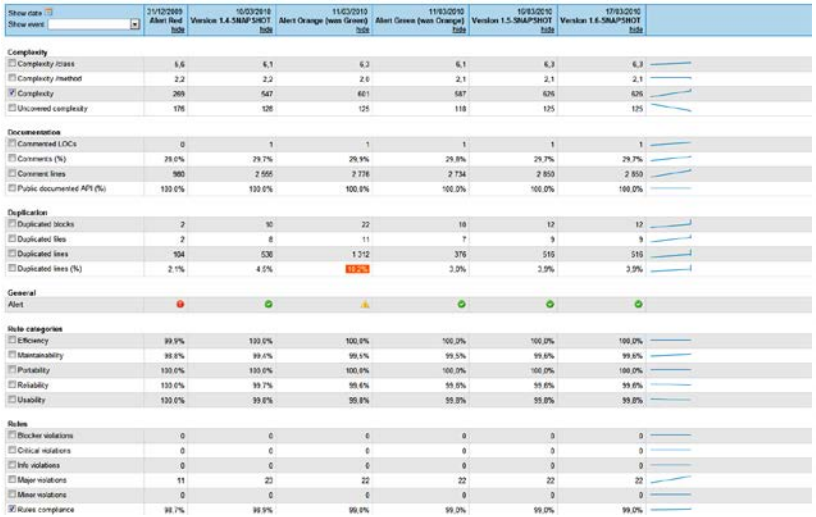
Document	Description
Checkstyle	Rapport sur les conventions de style de codage
Cobertura Test Coverage	Cobertura Test Coverage Report.
CPD Report	Duplicate code detection.
FindBugs Report	Generates a source code report with the FindBugs Library.
JavaDocs	JavaDoc API documentation.
JavaNCSS Report	Code metric analysis.
JDepend	JDepend traverses Java class file directories and generates design quality metrics for each Java package. JDepend allows you to automatically measure the quality of a design in terms of its extensibility, reusability, and maintainability to manage package dependencies effectively.
Liste des tags	Rapport sur les occurrences de tags trouvés dans le code.
Maven Surefire Report	Report on the test results of the project.
Rapport PMD	Vérification des règles de codage.
Simian report	Simian report of code duplications (may be similitudes too)
Sources Xref	Code source Java généré au format HTML.
Sources Xref des Tests	Code source Java pour les tests généré au format HTML.
Test JavaDocs	Test JavaDoc API documentation.

© 2008-2009 Netapsys

Les outils utilisés

- Sonar – Suivi qualité





Sonar

Home | Search | Scanbug »

- Dashboard
- Components
- Violations drilldown
- Time machine
- Clouds
- Hotspots

sonar

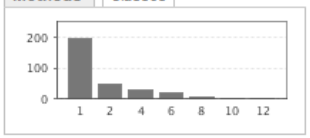
Version 1.6-SNAPSHOT - 17 mars 2010 01:32 - profile Netapsys

Lines of code
6 743
13 273 lines

Classes
99
30 packages
299 methods
+257 accessors

Complexity
2,1 / method
6,3 / class
626 cmpx
1 930 statements

Methods | Classes



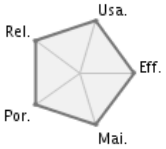
Comments
29,7%
2 850 lines
100,0% docu. API
1 commented LOCs

Duplications
3,9%
516 lines
12 blocks
9 files

Code coverage
80,0%
81,9% line coverage
70,0% branch coverage
151 tests
6.3 sec

Test success
100,0%
0 failures
0 errors

Rules compliance
99,0%



Violations
22

Blocker	0
Critical	0
Major	22
Minor	0
Info	0

No alerts.

Events All

17/03/2010	Version	1.6-SNAPSHOT
16/03/2010	Version	1.5-SNAPSHOT
11/03/2010	Alert	Green (was Orange)
11/03/2010	Alert	Orange (was Green)
10/03/2010	Version	1.4-SNAPSHOT
08/03/2010	Alert	Green (was Red)
01/03/2010	Version	1.3-SNAPSHOT
02/02/2010	Version	1.2-SNAPSHOT
01/02/2010	Version	1.1-SNAPSHOT
28/01/2010	Version	1.0-SNAPSHOT

Show more

Sonar

Show date / Show event	31/12/2009 Alert Red hide	10/03/2010 Version 1.4-SNAPSHOT hide	11/03/2010 Alert Orange (was Green) hide	11/03/2010 Alert Green (was Orange) hide	16/03/2010 Version 1.5-SNAPSHOT hide	17/03/2010 Version 1.6-SNAPSHOT hide
Complexity						
Complexity /class	5,6	6,1	6,3	6,1	6,3	6,3
Complexity /method	2,2	2,2	2,0	2,1	2,1	2,1
Complexity	269	547	601	587	626	626
Uncovered complexity	176	128	125	118	125	125
Documentation						
Commented LOCs	0	1	1	1	1	1
Comments (%)	28,0%	29,7%	29,9%	29,8%	29,7%	29,7%
Comment lines	980	2 555	2 776	2 734	2 850	2 850
Public documented API (%)	100,0%	100,0%	100,0%	100,0%	100,0%	100,0%
Duplication						
Duplicated blocks	2	10	22	10	12	12
Duplicated files	2	8	11	7	9	9
Duplicated lines	104	538	1 312	376	516	516
Duplicated lines (%)	2,1%	4,5%	10,2%	3,0%	3,9%	3,9%
General						
Alert	!	✓	⚠	✓	✓	✓
Rule categories						
Efficiency	99,9%	100,0%	100,0%	100,0%	100,0%	100,0%
Maintainability	98,8%	99,4%	99,5%	99,5%	99,6%	99,6%
Portability	100,0%	100,0%	100,0%	100,0%	100,0%	100,0%
Reliability	100,0%	99,7%	99,6%	99,6%	99,6%	99,6%
Usability	100,0%	99,8%	99,8%	99,8%	99,8%	99,8%
Rules						
Blocker violations	0	0	0	0	0	0
Critical violations	0	0	0	0	0	0
Info violations	0	0	0	0	0	0
Major violations	11	23	22	22	22	22
Minor violations	0	0	0	0	0	0
Rules compliance	98,7%	98,9%	99,0%	99,0%	99,0%	99,0%

Vision Développeur

- Réduction du temps de correction des bogues
  - On intègre nos modifications régulièrement
  - En cas de build en échec, la portion de code à débugger est faible
- Amélioration du travail collaboratif
  - La fin du « Ca marche sur mon poste pourtant ! »
  - Confiance entre le développeur et le logiciel
    - Le code et le design de l'application répondent aux exigences des standards, le résultat du build est un produit complètement fonctionnel et testable
- L'IC encourage des bonnes habitudes de test

Vision Développeur

- 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 »,…
  - Taux de couverture
  - Taux de duplication
  - Respect des conformités de Checkstyle

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 génère une compilation de tous ces rapports en prenant en compte la notion d'historique

Vision Chef de projet

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

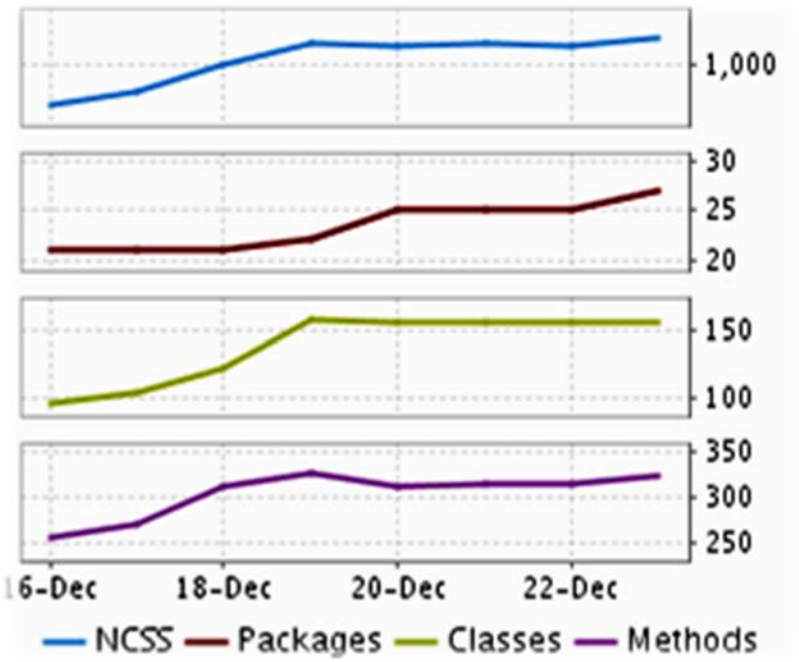
- L'objectif de la construction d'intégration est de produire un logiciel exécutable qui peut être déployé et testé fonctionnellement. Une démonstration avec le client est toujours possible

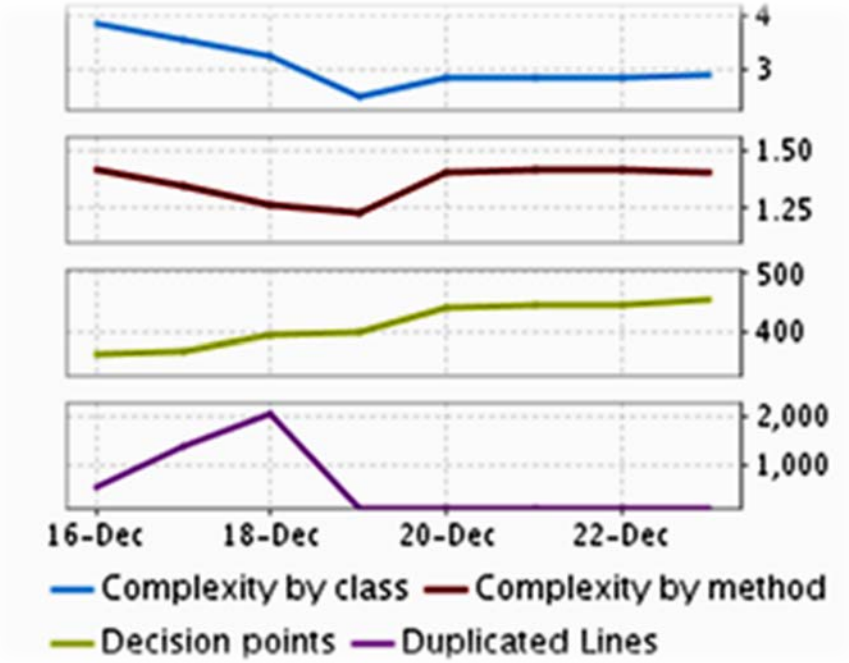
- Les outils d'assurance qualité participent au contrôle des risques :
  - Risque de faible qualité logiciel
  - Risque de découverte tardive des défauts

Vision Chef de projet

- Sonar, un outil central
  - Permet d'avoir une meilleure visibilité du projet. Il participe à la rétroaction
  - Synthétise les indicateurs d'assurance qualité produits par la construction Maven et fournit une vision chronologique de l'évolution du logiciel
  - Sonar est le tableau de bord de l'état technique et qualitatif de l'ensemble des projets de l'entreprise

Exemple de graphiques





Vision Chef de projet

- Pilotage / Suivi
  - L'intégration continue est devenue un outil fondamental pour le pilotage de nos projets : elle nous apporte plus de visibilité
- Qualité / Productivité
  - L'intégration continue est un formidable levier d'amélioration de notre niveau d'expertise technique : elle est l'un des moteurs de nos gains en termes de qualité et de productivité
- Transparence / Rétroaction
  - L'intégration continue est un élément différenciant par rapport à nos concurrents : elle est révélatrice de notre positionnement stratégique

Au sein de Netapsys

- Intégration continue mis en place depuis deux ans sur l'ensemble des projets.
- Tous les nouveaux projets sont systématiquement « mavenisés » et mis sous Hudson.
- Une fois mise en place, l'intégration continue n'est pas quelque chose de « figée »
- Evolution sur l'intégration continue en 2010 :
  - Actuellement: tests unitaires avec JUnit sur les projets
  - Mise en place de tests intégration et scénarios métier avec Sélénium

Le mot de la fin

- Pour aller plus loin
  - http://martinfowler.com/articles/continuousIntegration.html
  - https://hudson.dev.java.net/
  - http://sonar.codehaus.org/
  - http://maven.apache.org/
  - Vous pouvez consulter un dossier sur l'IC sur mon blog technique :
    - http://blog.fabianpiau.com/2009/07/17/continuous-integration/
- Vos questions