

Conduite de Projet - 2019/2020

Industrialisation dans le développement logiciel

Guillaume Kulakowski

Architecte Technique en solutions open-source @CGI



<https://twitter.com/llauogui>



<https://www.linkedin.com/in/guillaumekulakowski>



guillaume@kulakowski.fr



<https://blog.kulakowski.fr>

CGI



UNIVERSITÉ
DE MONTPELLIER

Agenda

1. Présentation.
2. Plan.
3. Code : Les VCS.
4. Build, Test & Release :
Plateforme d'Intégration
Continue.
5. Deploy : Déploiement Continu.
6. Operate.
7. Monitor.
8. Outils divers.

1 - Présentation

- Présentation de l'intervenant.
- Présentation du cours.

Le DevOps nouveau nom et nouvelle référence

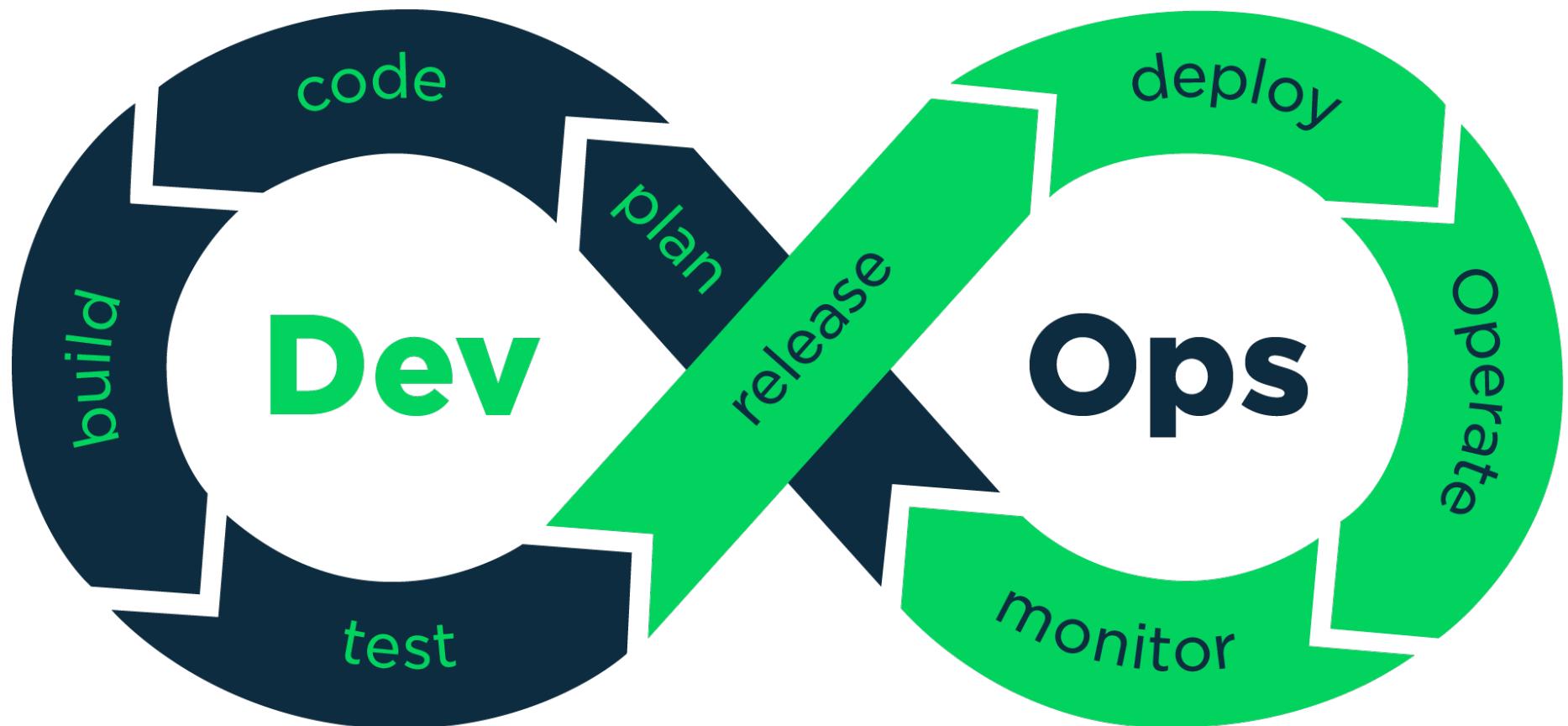
Le devops est un mouvement visant à l'alignement de l'ensemble des équipes du système d'information sur un objectif commun, à commencer par les équipes de dev ou dev engineers chargés de faire évoluer le système d'information et les ops ou ops engineers responsables des infrastructures (exploitants, administrateurs système, réseau, bases de données,...).

Ce qui peut être résumé par : travailler ensemble pour produire de la valeur pour l'entreprise.

Le DevOps nouveau nom et nouvelle référence dans l'industrialisation du développement logiciel

©Wikipédia

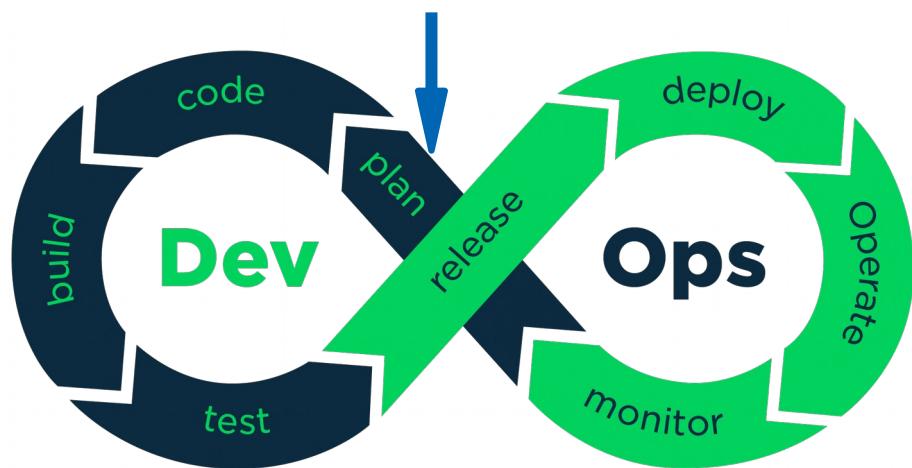
Une représentation qui semble faire l'unanimité



2 – Plan

- Définition.
- Exemple.

Le cycle DevOps



Les outils de planification

Planifier est composé en fait en deux parties : « **Définir** » et « **planifier** ». Cette étape concerne la valeur commerciale et les exigences des applications. Plus précisément, les activités "Plan" comprennent :

- Indicateurs, objets, et retour d'expérience de production
- Exigences
- Mesures commerciales
- Métriques de mise à jour et de mise en production
- Plan, timing et analyse de rentabilisation des releases
- Politique de sécurité

TIS-70 Scrum Board		
QUICK FILTERS: Critical partners Only my partners Recently updated		
12 To do	2 In progress	3 Done
TIS Developer Love 3 issues		
TIS-37 Service should return prior trip details and info <small>SeeSpaceEZ plus</small>	TIS-10 Bad JSON data coming back from hotel API <small>SeeSpaceEZ plus</small>	TIS-8 Requesting flights is now taking > 5 seconds <small>SeeSpaceEZ plus</small>
Everything Else 21 issues		
TIS-68 Homepage footer uses an inline style-should use class <small>Large Team Support</small>	TIS-17 Engage Saturn's Rings Resort as preferred <small>Space Travel Partners</small>	TIS-56 Add pointer to main css file to create child themes <small>Large Team Support</small>
TIS-20 Engage Saturn Shuttle lines for group tours <small>Space Travel Partners</small>	TIS-12 Create 90 day plans for all departments in Mars office <small>SeeSpaceEZ plus</small>	TIS-45 Email non registered users to sign up with TIS <small>SeeSpaceEZ plus</small>

Une utilisation classique : *bugtracker*

- Malgré tout notre outillage, il est ~~possible~~ probable qu'un (ou plusieurs) bug(s) ai(en)t échappé(s) à notre vigilance.
- L'outil de planning sert alors au ticketing pour renseigner les incidents.
- Chaque outil possède son workflow.
- Dans le cadre d'une anomalie, il ne faut pas oublier que la première étape est toujours la **qualification** :
 - Est-ce vraiment une anomalie ?
 - Est-ce une évolution ?
 - Est-ce une anomalie en garantie ?
 - Etc...

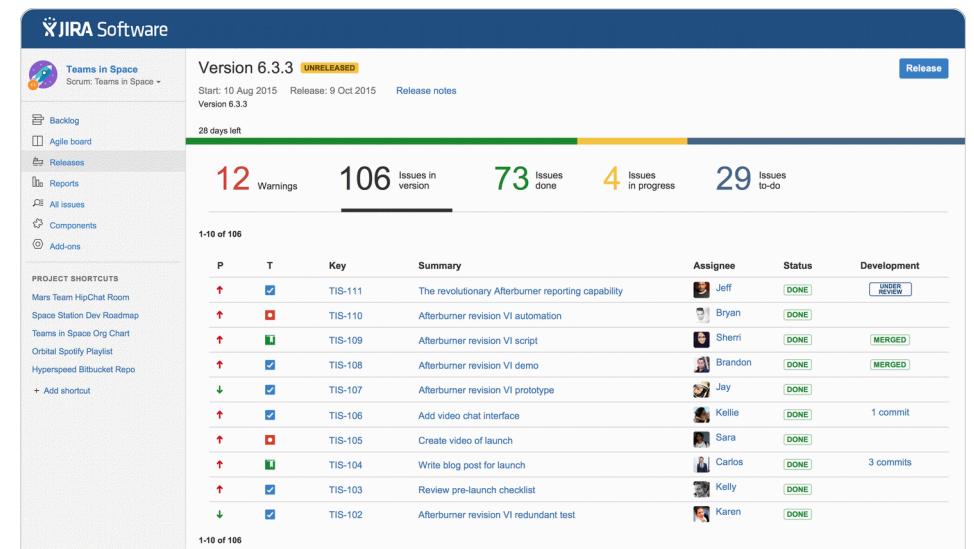


Diagram Text Export ▾

+ Add status

+ Add transition

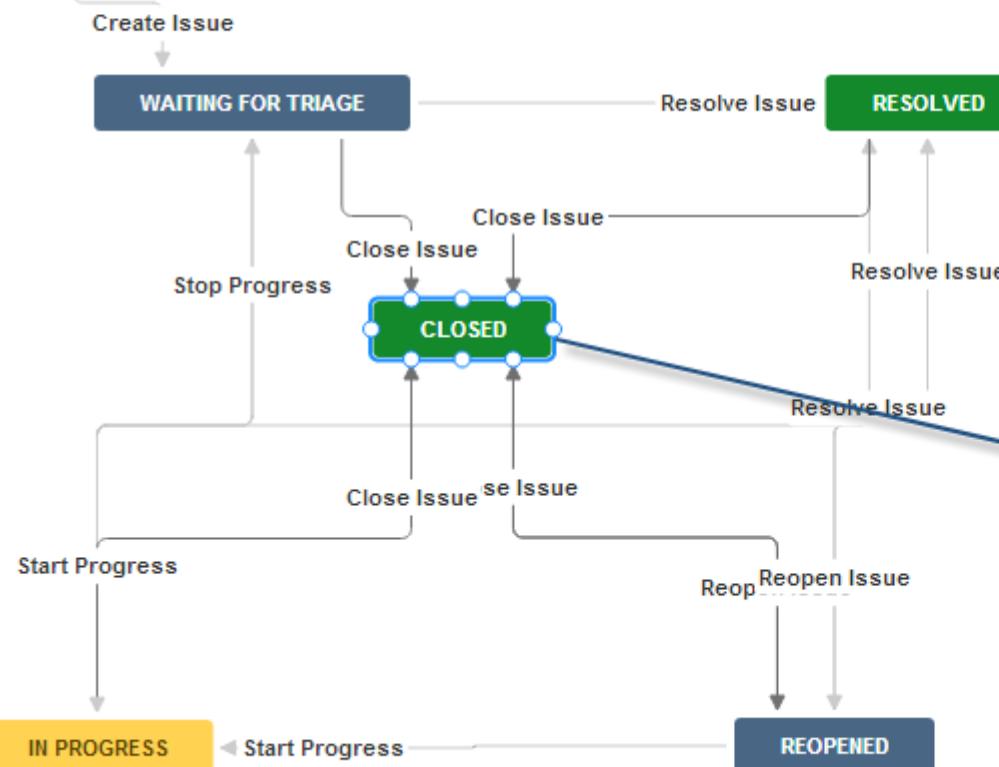
Show transition labels

Last edited by you, Yesterday 3:06 PM



Add a new/existing status to the workflow

Add a new/existing transition to the workflow



Closed

Description The issue is considered finished, the resolution is correct. Issues w/ ... [Show more](#)

Allow all statuses to transition to this one



Edit Remove status

Options

Properties

Click a status in the diagram to show this panel

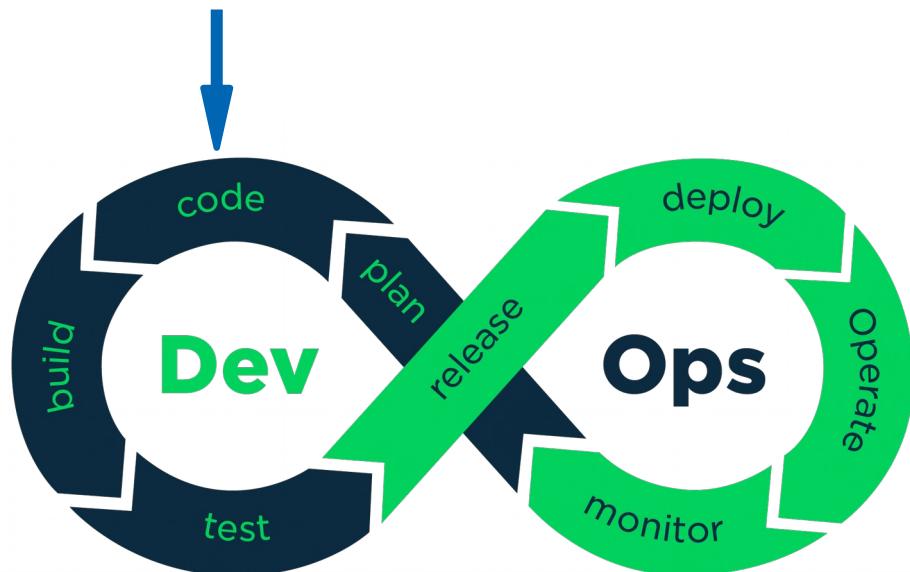
ABANDONED

All

3 – Code: Les VCS

- Définition.
- Les différents types de VCS.
- Focus sur GIT.
- Les workflows.

Le cycle DevOps



Définition

Un **logiciel de gestion de versions** (ou **VCS** en anglais, pour *Version Control System*) est un logiciel qui permet de stocker un ensemble de fichiers en conservant la chronologie de toutes les modifications qui ont été effectuées dessus. Il permet notamment de retrouver les différentes versions d'un lot de fichiers connexes.

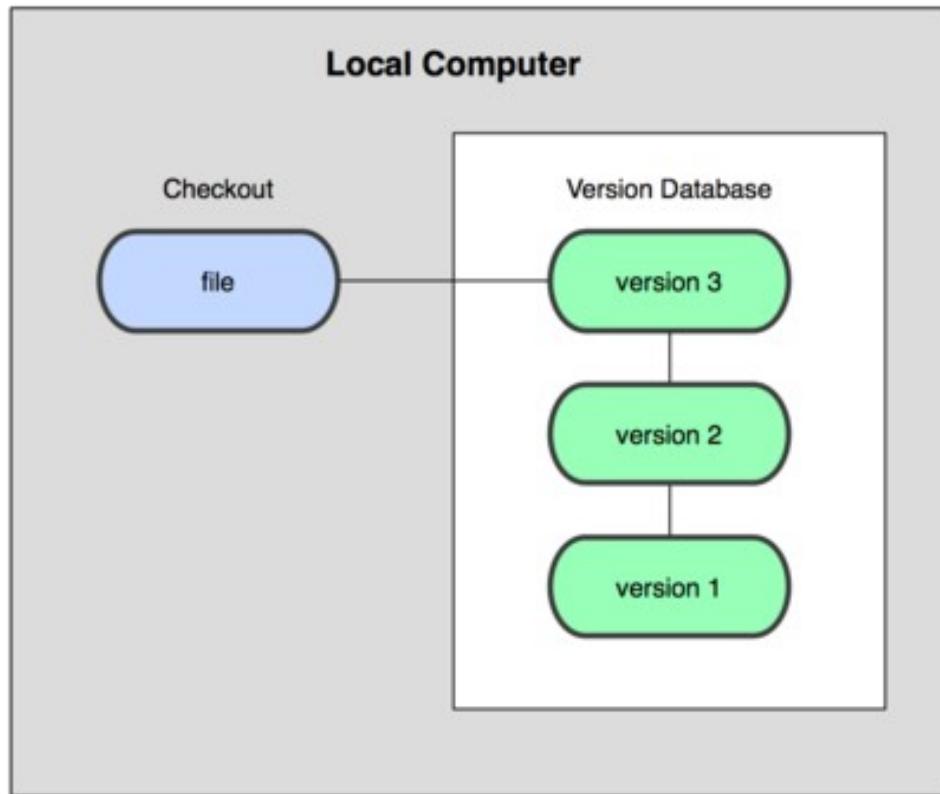
Il existe aussi des logiciels et services de gestion de versions **décentralisé** (distribué) (ou **DVCS** en anglais, pour *Distributed Version Control System*). Git et Mercurial sont deux exemples de logiciel de gestion de versions décentralisé et sont disponibles sur la plupart des systèmes Unix et Windows.

Les logiciels de gestion de versions sont utilisés notamment en développement logiciel pour conserver le code source relatif aux différentes versions d'un logiciel. (© Wikipedia)

Répond à des besoins

- **Partager** des sources en équipe.
- **Partager** des sources avec tout le monde (open source).
- **Historique** des sources.
- **Tuiler**, c'est à dire gérer plusieurs phase (branches) ou versions d'un projet en parallèle.

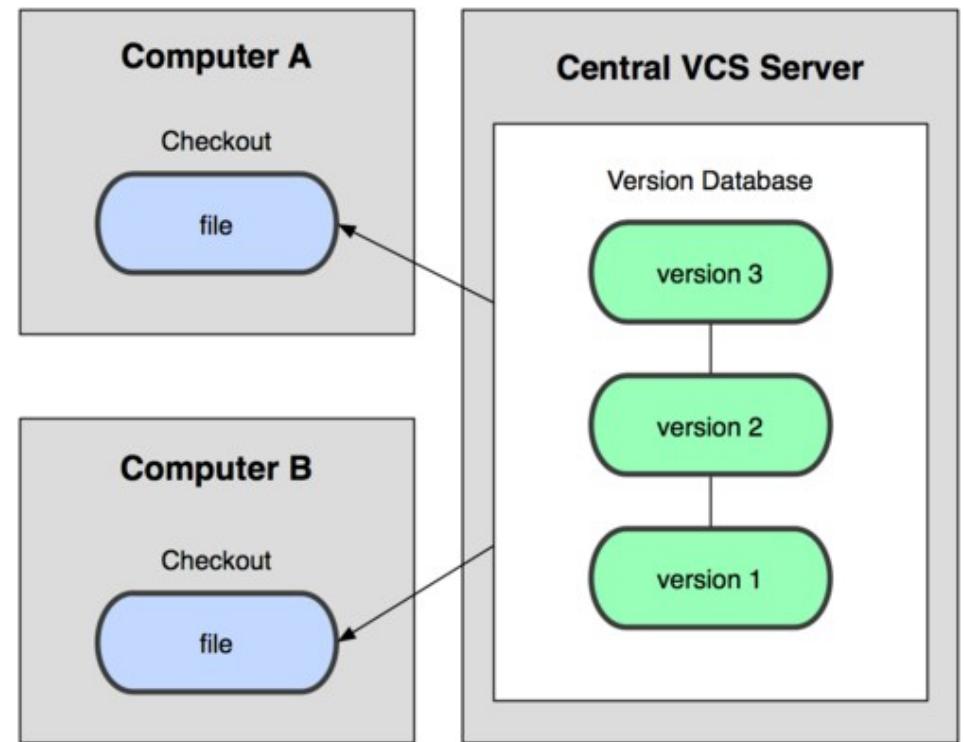
Type #1 : Gestion de versions locale



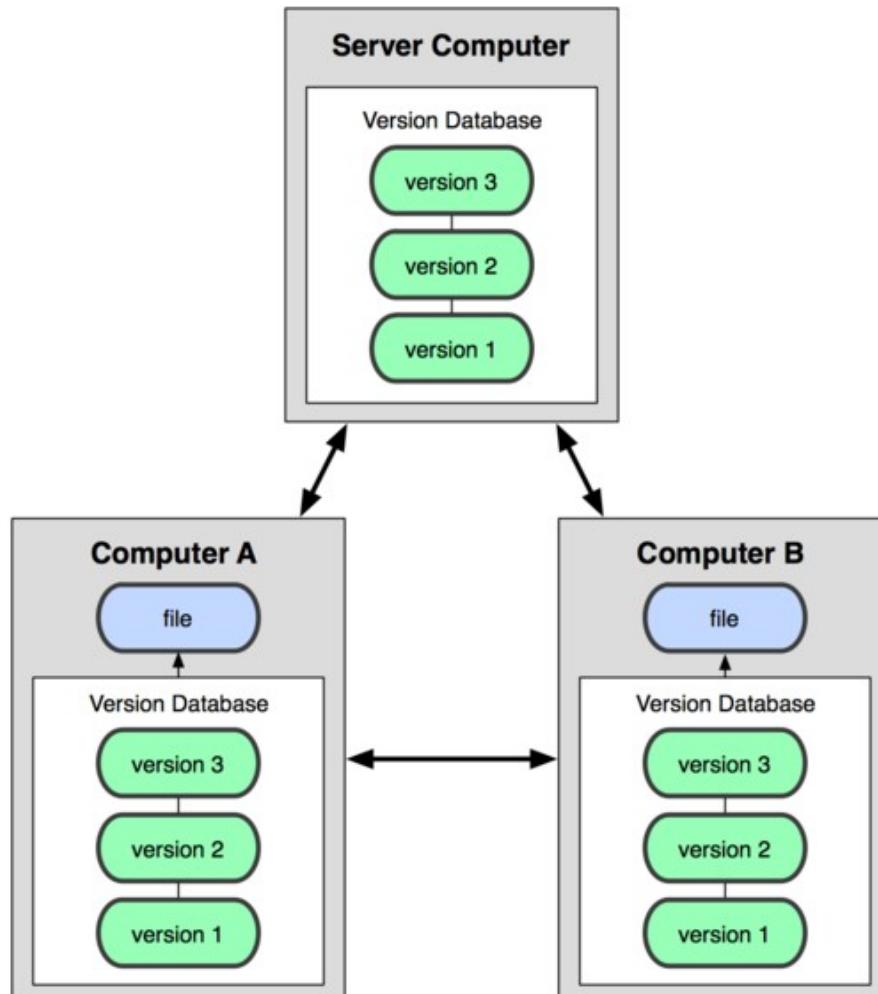
- Exemple : [RCS](#)
- Système plus répandu (heureusement) de nos jours.
- Ce schéma ressemble à ce que vous propose vos IDE avec l'historique local.

Type #2 : Gestion de versions centralisée

- VSC.
- Exemple [SVN](#).
- Tout passe par le serveur central : [SPOF](#).
- Les versions locales (et donc poussées) sont anarchiques et incohérentes (possibilité d'*update* une arborescence partielle).



Type #3 : Gestion de versions décentralisée



- DVSC.
- Exemple [GIT](#), [Hg](#), [Bz](#), etc.
- Serveur local clone du distant (rapidité).
- Les versions locales (et donc poussées) sont cohérentes car on ne peut pousser si le dépôt n'est pas en phase.
- Puissant donc complexe à utiliser.
- Vraie gestion de branche.

Focus sur GIT : Historique

- Une histoire liée au noyau **Linux** et à **Linus Torvalds**.
- En **2002**, le projet du noyau Linux commence à utiliser un DVCS propriétaire : BitKeeper.
- En **2005**, clash entre la communauté & BitKeeper qui passe sur un modèle payant.
- Suite à ça, toujours en **2005**, La réponse de la communauté & de Linus fut la **création de GIT**.

Focus sur GIT : ADN

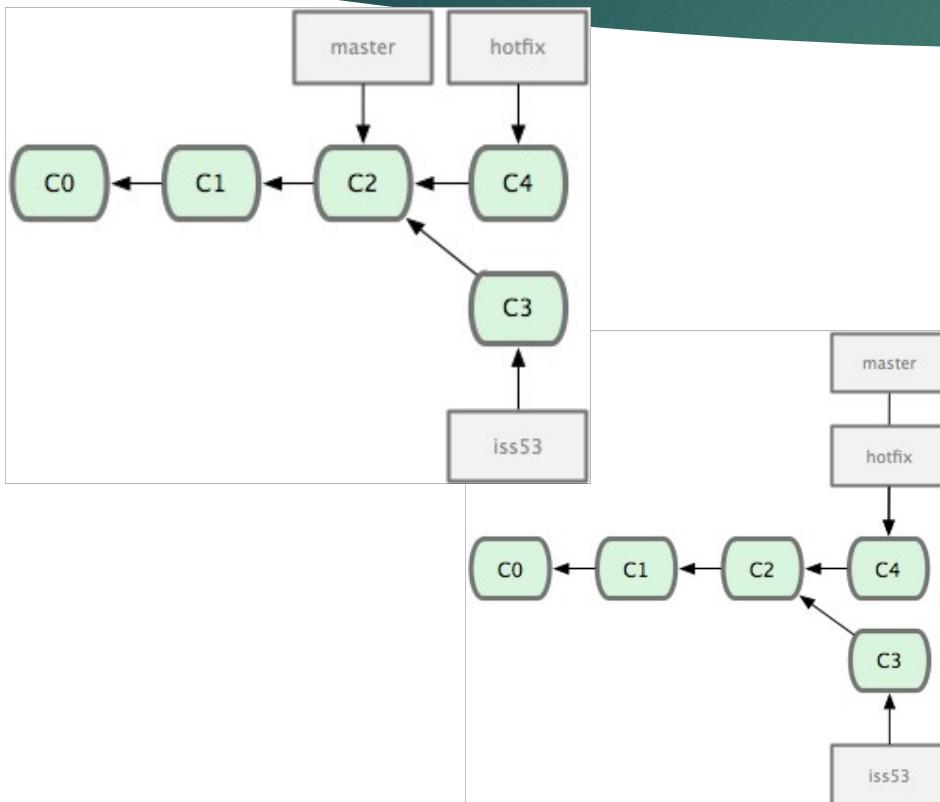
Le cahier des charges :

- **Vitesse.**
- Conception **simple**.
- Support pour les développements non linéaires (**milliers de branches** parallèles).
- Complètement **distribué**.
- Capacité à **gérer efficacement des projets d'envergure** tels que le noyau Linux (vitesse et compacité des données).

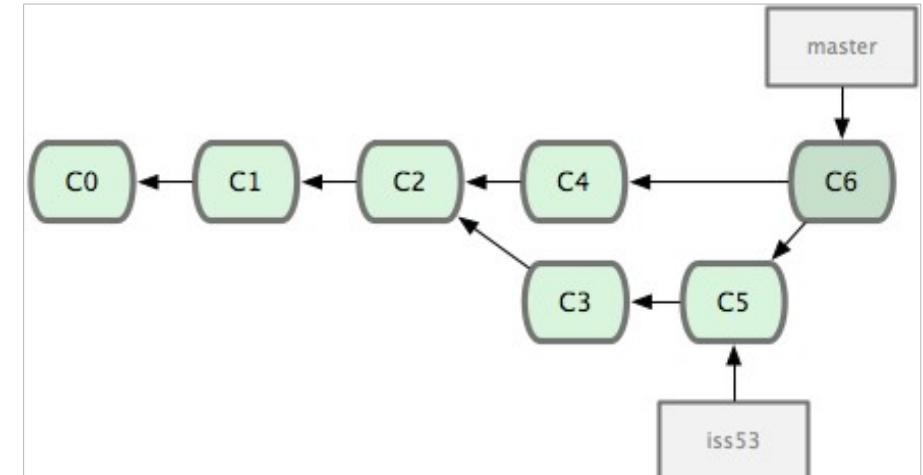
Focus sur GIT : les killer features

- **rebase** : introduire les commit d'une autre branche.
- **reset** : annuler des commits non poussés.
- **revert** : retour en arrière.
- **amend** : amender le dernier commit.
- **bisect** : localiser le commit foireux par dichotomie.
- **stash** : remiser votre code.
- **squash** : fusion de commits.

2 types de merges



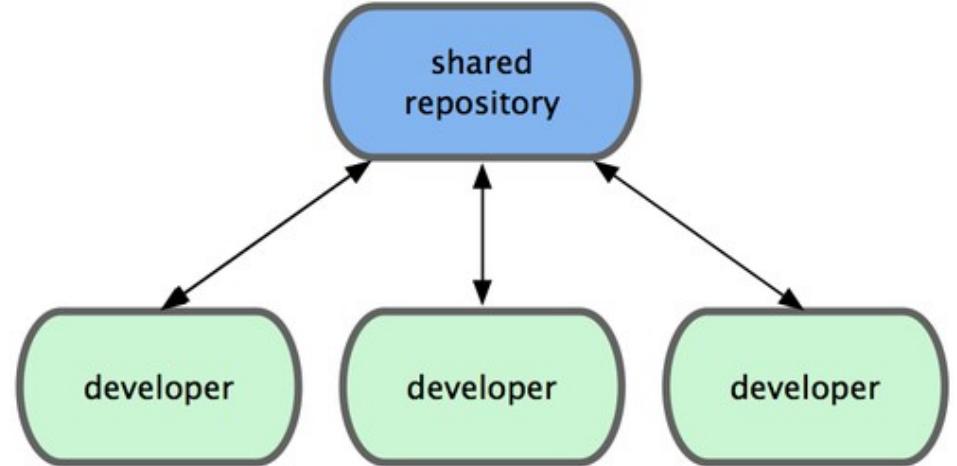
Le merge dit "fast forward" est une simple avance rapide, un déplacement de pointeur.



Le merge classique avec création d'un commit de merge.

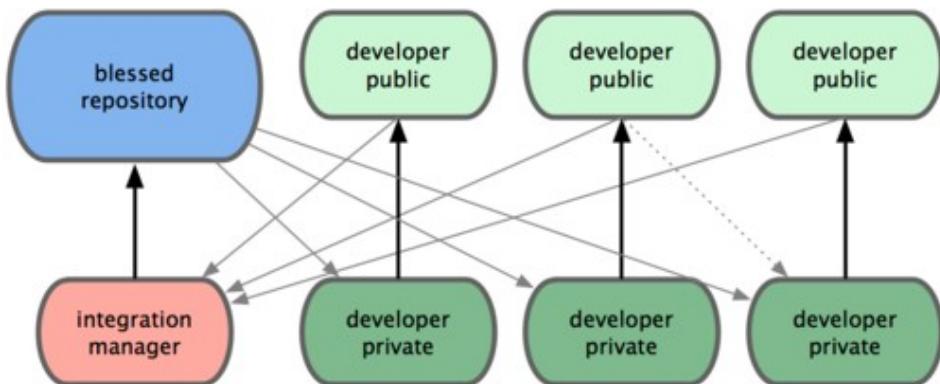
Workflow #1 : Mode distribué

- Le plus utilisé car le plus simple.
- Proche du VSC.
- Tous les développeurs *push* sur le même dépôt distant.



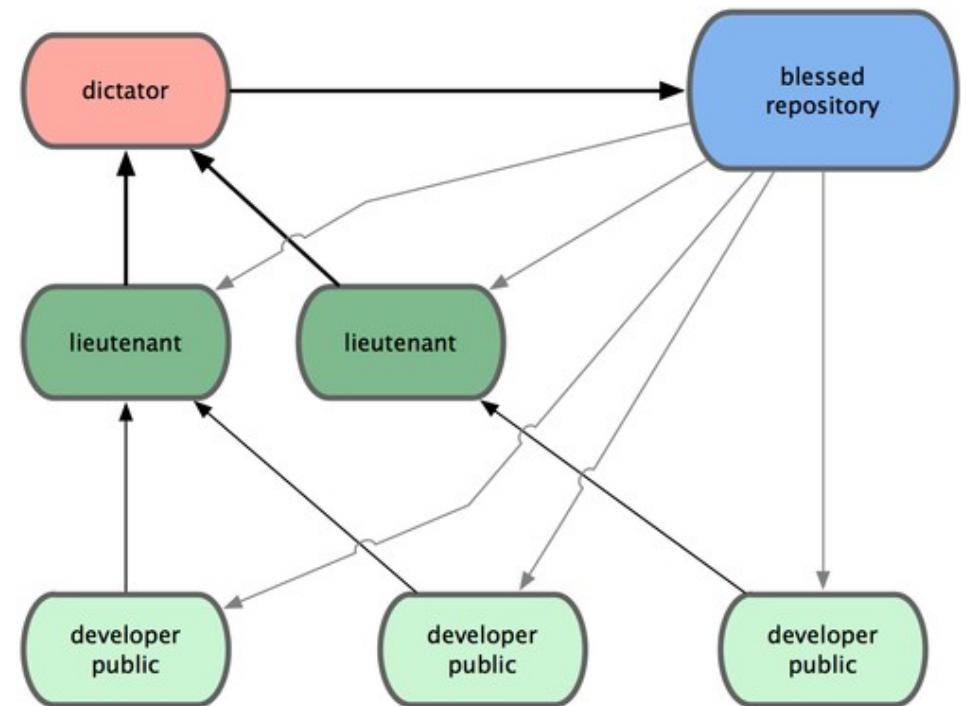
Workflow #2 : Gestionnaire d'intégration

- Approche proche du principe de *Pull Request* de [GitHub](#).
- 1 gardien du temple vérifie les *PR* avant de les *merger* dans le dépôt officiel.



Workflow #3 : Dictateur

- La méthode la plus efficace pour gérer les gros projets subdivisés en plusieurs sous-projets eux même subdivisés en d'autres projets...
- Exemple Linux.



Workflow #4 : GIT-Flow

- Plus une méthode de gestion de branches qu'un workflow.
- Utilisation de branches sémantiques :
 - **master** : code de production. On ne développe rien sur cette branche.
 - **develop** : code prêt pour le *merge* pour la nouvelle release. On peut également corriger, améliorer de petites fonctionnalités.
 - **feature/xxx** : développement des nouvelles fonctionnalités. Une fois terminé, on *merge* les changements dans *develop*.
 - **release/vx.y.z** : Préparation d'une nouvelle release. Un *merge* sera alors effectué dans *master*.
 - **hotfix** : permet de réparer tout de suite un bug critique en production.

Gérer votre projet et votre VCS via une forge : Exemple de GitLab

The screenshot shows a GitLab repository page for the project 'blog.kulakowski-fr'. The left sidebar contains navigation links for Overview, Repository (selected), Fichiers, Validations, Branches, Tags, Contributeurs, Graphes, Comparer, and Statistiques. It also shows 0 issues and settings. The main content area displays the 'Repository' page for the 'develop' branch. The README.md file is shown with its content: 'Boldy Dotclear Theme' and a bullet point about project management and GIT repository. Below this is an 'Install' section with two steps: 'Get all Nodejs tools and dependencies' (with a command line entry for 'npm install') and 'Use Grunt to build CSS and JavaScript files' (with a command line entry for 'grunt build'). There is also a 'Remark' section with a bullet point about using the 'dev' task to watch Less and build theme on the fly, followed by a command line entry for 'grunt dev'. The 'Deploy' section contains instructions to copy ./deployrc.dist to ./deployrc and use grunt to deploy in production, with a command line entry for 'grunt deploy'. At the bottom, there is a 'Copyright and license' section stating 'Theme by Site5 (<http://www.s5themes.com>), under a GPLv2. Ported on Dotclear by Guillaume Kulakowski.'

Aller plus loin...

Git, GitHub & social coding

Par Guillaume KULAKOWSKI

Expert Technique en Solutions Open Source @ [CGI](#).

Ambassadeur & Packageur @ [FedoraProject](#).

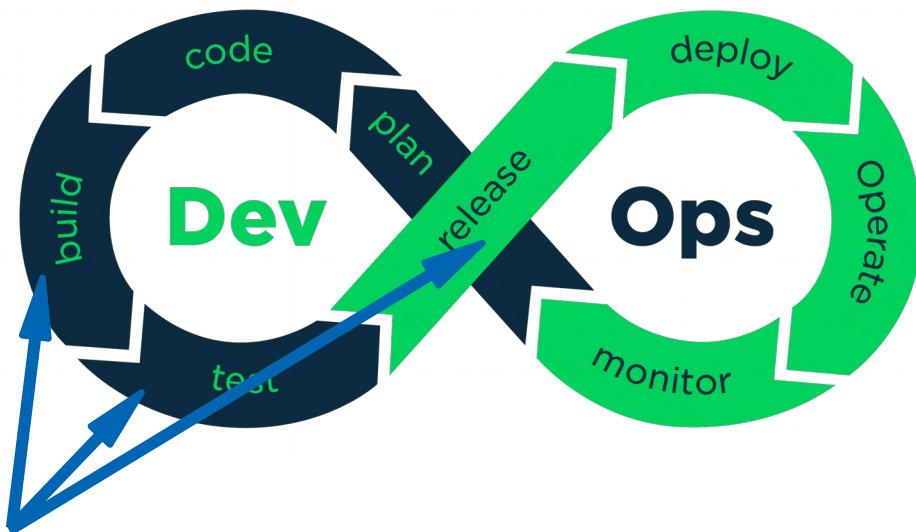
Blogueur @ [blog.kulakowski.fr](#).



4 – Build, Test & Release Plateforme d'Intégration Continue

- Définition
- Exemple
- Outil de qualimétrie
- Métriques

Le cycle DevOps



Jenkins



maven



COMPOSER



Définition

L'intégration continue est un ensemble de pratiques utilisées en génie logiciel consistant à vérifier à chaque modification de code source que le résultat des modifications ne produit pas de régression dans l'application développée.

Le concept a pour la première fois été mentionné par Grady Booch et se réfère généralement à la pratique de l'extreme programming. Le principal but de cette pratique est de détecter les problèmes d'intégration au plus tôt lors du développement. De plus, elle permet d'automatiser l'exécution des suites de tests et de voir l'évolution du développement du logiciel.

L'intégration continue est de plus en plus utilisée en entreprise afin d'améliorer la qualité du code et du produit final.(© Wikipedia)

Concrètement ?

Un PIC ça sert à quoi ?

- A construire (build, compilation, etc.).
- A générer la documentation.
- A lancer des **tests de qualité** (QA, qualimétrie).
- A lancer des **tests d'intégration** (TI) automatisés.
- A lancer des **tests de non-régression** (TNR) automatisés.
- A lancer des **tests de unitaires** (TU) automatisés.
- A construire un **paquet** (et même à le déployer).
- A lancer des **outils divers** (tests, sondes, etc.).
- Etc...

Le rôle central de l'ordonnanceur

Comme le montre le choix du logo de Jenkins, une PIC ne fait pas grande chose, elle se base sur un ordonnanceur pour faire le travail.

- [ANT / Maven / Gradle](#) : ordonnanceur JAVA.
- [Phing / Composer](#) : ordonnanceur PHP.
- Etc...



C'est eux qui au travers de leur langage de script vont faire les appels aux outils (tests, qualimétrie, etc.).

Exemple de PIC #1 : Travis

This screenshot shows the GitHub repository page for `llaumgui/JunitXml`. At the top, there are standard GitHub navigation links: Pull requests, Issues, Gist, Unwatch, Unstar, Fork, and a search bar. Below the header, the repository name `JunitXml` is displayed, along with its description: "PHP library for generate XML document in JUnit format." A prominent red box highlights the build status bar at the top of the repository page, which includes buttons for build (green), code climate (4.0), Snyk (green), coverage (100%), and php package (0.2.0).

Documentation

- For documentation, see [JunitXml documentation](#).
- For API documentation, see [JunitXml API documentation](#).

Quick how to install

With `composer`, run:

```
$ php composer.phar require llaumgui/junit-xml
```

License

Released under the [MIT license](#).

This screenshot shows the Travis CI build history for the `llaumgui/JunitXml` repository. The main interface includes a search bar, a sidebar for "My Repositories" (with `llaumgui/JunitXml` listed), and tabs for Current, Branches, Build History, and Pull Requests. The "Build History" tab is active, showing a list of recent builds.

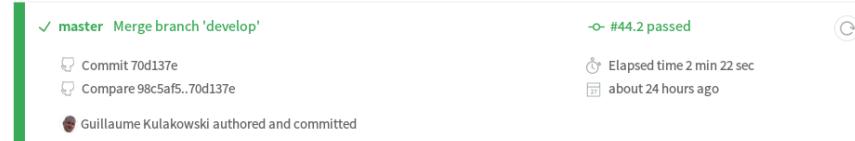
Build ID	Status	Commit	Duration	Author	Last Updated
#44	Passed	70d137e	2 min 22 sec	Guillaume Kulakowski	about 24 hours ago
#68	Pending	98c5af5..70d137e	1 min 43 sec	Guillaume Kulakowski	5 days ago
#36	Pending	70d137e	2 min 26 sec	Guillaume Kulakowski	8 days ago
#63	Pending	70d137e	23 sec	Guillaume Kulakowski	14 days ago

Build Jobs

ID	Environment	Variables	Duration
# 44.1	PHP: 5.5	No environment variables set	2 min 9 sec
# 44.2	PHP: 5.6	No environment variables set	2 min 22 sec
# 44.3	PHP: 7	No environment variables set	2 min 1 sec

Allowed Failures

ID	Environment	Variables	Duration
# 44.4	PHP: hhvm	No environment variables set	1 min 49 sec
# 44.5	PHP: nightly	No environment variables set	1 min 49 sec



Using worker: worker-linux-docker-9ec0601a.prod.travis-ci.org:travis-linux-1

Build system information

```
1 Using worker: worker-linux-docker-9ec0601a.prod.travis-ci.org:travis-linux-1
2
3 Build system information
4
5 67
6 68 $ export DEBIAN_FRONTEND=noninteractive
7 69
8 70 $ git clone --depth=50 --branch=master https://github.com/ltaumgui/JunitXml.git
9 71
10 72 This job is running on container-based infrastructure, which does not allow use of 'sudo', setuid and setgid
11 73 executables.
12 74 If you require sudo, add 'sudo: required' to your .travis.yml
13 75 See https://docs.travis-ci.com/user/workers/container-based-infrastructure/ for details.
14 76
15 77 Setting environment variables from repository settings
16 78
17 79 $ export CODECLIMATE_REPO_TOKEN=[secure]
18 80 $ export GH_TOKEN=[secure]
19 81
20 82
21 83 $ phpenv global 5.6 2>/dev/null
22 84
23 85
24 86 $ php --version
25 87 PHP 5.6.5 (cli) (built: Feb 12 2015 01:41:10)
26 88 Copyright (c) 1997-2014 The PHP Group
27 89 Zend Engine v2.6.0, Copyright (c) 1998-2014 Zend Technologies
28 90     with Zend OPcache v7.0.4-dev, Copyright (c) 1999-2014, by Zend Technologies
29 91     with Xdebug v2.2.7, Copyright (c) 2002-2015, by Derick Rethans
30 92
31 93 $ composer --version
32 94 Warning: This development build of composer is over 30 days old. It is recommended to update it by running "/home/travis
33 95 /phpenv/versions/5.6/bin/composer self-update" to get the latest version.
34 96 Composer version 1.0.0-dev (1d9ff05f1dd0e390f253f79ea86cd505178360019) 2015-02-11 11:31:57
35 97
36 98
37 99
40 100 $ composer self-update
41 101 $ composer install --prefer-source
42 102 $ ./vendor/bin/phpunit --configuration phpunit.xml.dist --coverage-text --coverage-clover build/logs/clover.xml
43 103 --coverage-html build/coverage
44 104
45 105 PHPUnit 4.8.23 by Sebastian Bergmann and contributors.
46 106
47 107
48 108
49 109
50 110
51 111
52 112
53 113 Time: 1.54 seconds, Memory: 9.75Mb
54 114
55 115
56 116
57 117 Generating code coverage report in Clover XML format ... done
58 118
59 119 Generating code coverage report in HTML format ... done
60 120
61 121
62 122 Code Coverage Report:
63 123 2016-03-01 19:42:21
64 124
65 125 Summary:
66 126
67 127
68 128
69 129
70 130 \Ltaumgui\JunitXml::JunitXmlTestCase
71 131
72 132 \Ltaumgui\JunitXml::JunitXmlTestElement
73 133
74 134 \Ltaumgui\JunitXml::JunitXmlTestSuite
75 135
76 136 \Ltaumgui\JunitXml::JunitXmlTestSuites
77 137
78 138 \Ltaumgui\JunitXml::JunitXmlValidation
79 139
80 140
81 141
82 142 The command "./vendor/bin/phpunit --configuration phpunit.xml.dist --coverage-text --coverage-clover build/logs
83 143 /clover.xml --coverage-html build/coverage" exited with 0.
84 144
85 145
86 146 The command "./vendor/bin/phpcs" exited with 0.
87 147
88 148
89 149
90 150
91 151
92 152
93 153
94 154 Done. Your build exited with 0.
```

X Remove log Raw log

Top ▲

lbaumgui / lesshint-lint-xml-reporter

Code Issues 0 Pull requests 0 Projects 0 Insights Settings

Branch: master lesshint-lint-xml-reporter / .travis.yml

lbaumgui Need NodeJS 4+ and fix travis versions

1 contributor

9 lines (8 sloc) | 78 Bytes Raw

```
1 language: node_js
2 node_js:
3   - "node"
4   - "8"
5   - "7"
6   - "6"
7   - "5"
8   - "4"
```



<https://github.com/lbaumgui/lesshint-lint-xml-reporter>

 This repository Search Pull requests Issues Marketplace Explore  + 

lbaumgui / lesshint-lint-xml-reporter

Code Issues 0 Pull requests 0 Projects 0 Insights Settings

Branch: master lesshint-lint-xml-reporter / package.json Find file Copy path

lbaumgui Update doc b63edbf on 16 Sep 2017

1 contributor

44 lines (43 sloc) | 1.03 KB Raw Blame History

```
1 {
2   "name": "lesshint-lint-xml-reporter",
3   "description": "A lesshint's reporter using the same lint-xml format that CSSLint. Can be used by Jenkins.",
4   "version": "1.0.0",
5   "main": "./reporter.js",
6   "homepage": "https://github.com/lbaumgui/lesshint-lint-xml-reporter",
7   "author": {
8     "name": "Guillaume Kulakowski",
9     "url": "http://blog.kulakowski.fr/"
10 },
11 "repository": {
12   "type": "git",
13   "url": "https://github.com/lbaumgui/lesshint-lint-xml-reporter.git"
14 },
15 "keywords": [
16   "less",
17   "lint",
18   "hint",
19   "lesshint",
20   "reporter"
21 ],
22 "engines": {
23   "node": ">=4.0.0"
24 },
25 "dependencies": {
26 },
27 "devDependencies": {
28   "grunt": "latest",
29   "grunt-eslint": "latest",
30   "grunt-travis-lint": "latest",
31   "grunt-jsonlint": "latest",
32   "grunt-coffeelint": "latest",
33   "grunt-mdlint": "latest"
34 },
35 "scripts": {
36   "test": "grunt travis --verbose",
37   "lint": "grunt lint --verbose"
38 },
39 "license": "MIT",
40 "bugs": {
41   "url": "https://github.com/lbaumgui/lesshint-lint-xml-reporter/issues"
42 }
43 }
```



✓ SmoovenGO / Front web 906

[Pipeline](#)[Modifications](#)[Tests](#)[Artefacts](#)

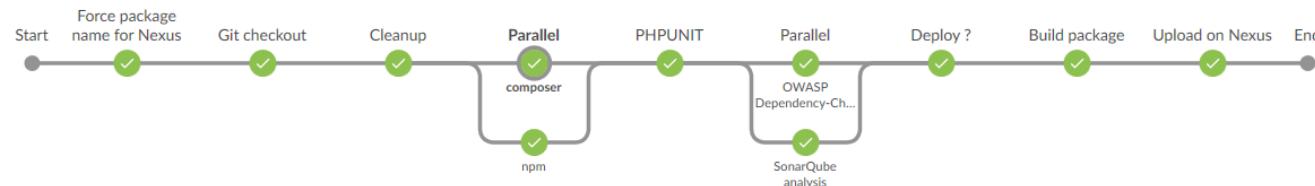
Déconnexion



Branche: develop
Commit: 1fae342

⌚ 11m 39s
⌚ an hour ago

Modifications par remy.bannes
Branch indexing



Parallel / composer - <1s



✓	> Shell Script	1m 11s
✓	> Shell Script	9s
1	[application] Running shell script	
2	+ composer/l ci	
3	> vendor/bin/phpcs --config-set installed_paths vendor/escapestudios/symfony2-coding-standard	
4	Using config file: /var/jenkins_home/workspace/smoovengo_front_web Develop-ODMHGB2A656HXWM10C4E3YURWVEZG66WUXYRWYZWU2ERZUIRIR2Q/application/vendor/squizlabs/php_codesniffer/CodeSniffer.conf	
5		
6	Config value "installed_paths" added successfully	
7	> vendor/bin/phpcs --standard=Symfony --ignore="*/tests/*,*/data/*,*/Tests/*" --report=checkstyle --report-file=../logs/checkstyle.xml src	
8	Time: 7.16 secs; Memory: 20Mb	
10	Script vendor/bin/phpcs --standard=Symfony --ignore="*/tests/*,*/data/*,*/Tests/*" --report=checkstyle --report-file=../logs/checkstyle.xml src handling the phpcs event returned with error code 2	
✓	> Publier les résultats de l'analyse Checkstyle	1s
✓	> injected_BRANCH="develop" composer71 sami — Shell Script	2s
✓	> Publish HTML reports	<1s

Métrique : outil de type « Checkstyle »

- Exemple : [Checkstyle](#) (JAVA), [PHPCS](#) (PHP), etc.
- Contrôle la cohérence du style d'un code source :
 - Choix des noms de classe et fonction,
 - Indentation,
 - Nom des fichiers,
 - Namespacing,
 - Etc...

Métrique : duplication de code

- Exemple [PHPCPD](#), [PMD](#)
- Permet de détecter les copier/coller et les manques de factorisation au sein d'un projet.

Métrique : complexité

- Exemple : [PMD](#), [PHPMD](#), [Pdepend](#), etc.
- Détection de code mort, fonctions et/ou variable inutilisé.
- Notion de complexité cyclomatique.
- Longueur méthodes, classes, etc...
- Etc...

```
// Cyclomatic Complexity = 12
class Foo {
1   public function example()  {
2     if ($a == $b)  {
3       if ($a1 == $b1) {
4         fiddle();
5       } else if ($a2 == $b2) {
6         fiddle();
7       } else {
8         fiddle();
9       }
10      } else if ($c == $d) {
11        while ($c == $d) {
12          fiddle();
13        }
14      } else if ($e == $f) {
15        for ($n = 0; $n < $h; $n++) {
16          fiddle();
17        }
18      } else{
19        switch ($z) {
20          case 1:
21            fiddle();
22            break;
23          case 2:
24            fiddle();
25            break;
26          case 3:
27            fiddle();
28            break;
29          default:
30            fiddle();
31            break;
32        }
33      }
34    }
35}
```



Métrique : sécurité

Jenkins - Nightly build (mail) #17 - Dependency-Check Warnings

Dependency-Check Result

Tendance des Alertes

Toutes les Alertes	Nouvelles Alertes	Alertes résolues
22	0	0

Résumé

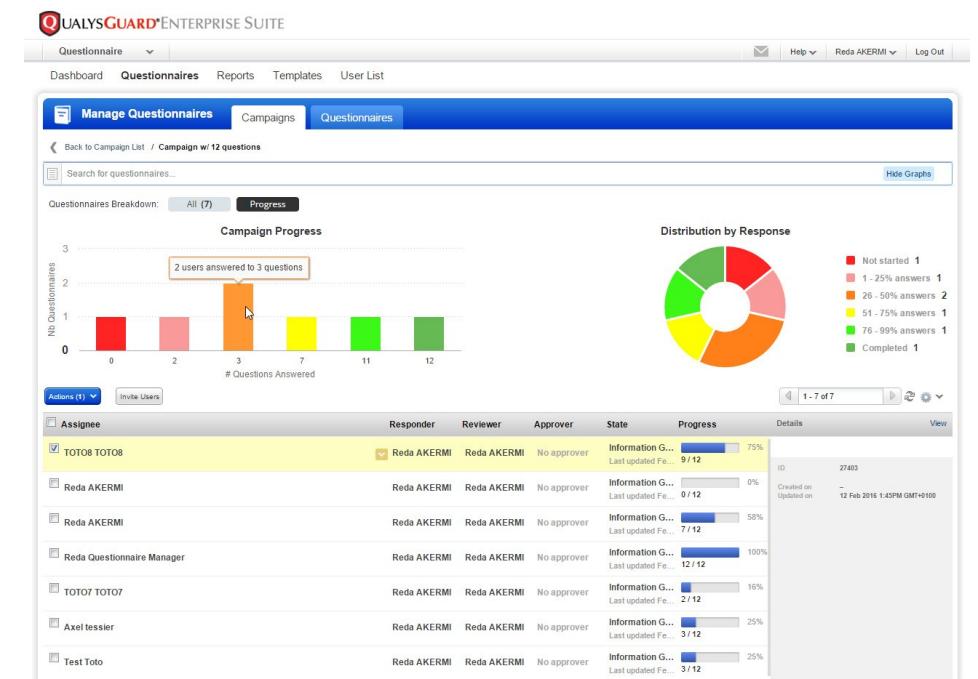
Total	Priorité haute	Priorité normale	Priorité basse
22	15	7	0

Détails

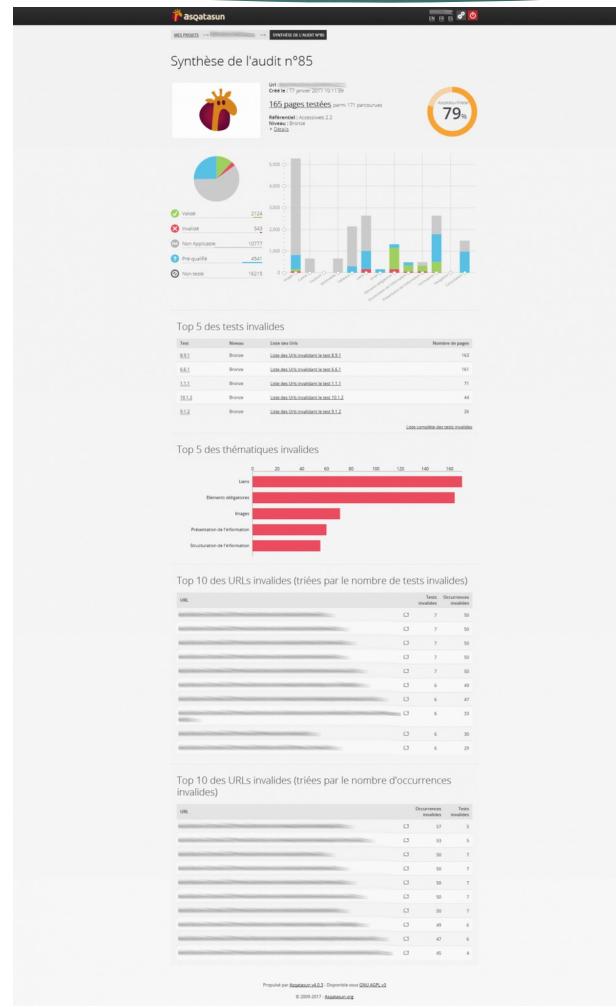
Source Folder	Total Distribution
varibitkenr/m2repository/com/testfarm/jackson/conversation-annotations/2.4.0	1
varibitkenr/m2repository/com/testfarm/jackson/conversation-concept/2.2	1
varibitkenr/m2repository/com/testfarm/jackson/datformat/jackson-datatype-yaml/2.4.2/jackson-datatype-yaml/2.4.2/jacksonTA/nfl/maven	1
com.fasterxml.jackson.datatype.jaxrs/2.4.2/jacksonTA/nfl/jaxrs	1
varibitkenr/m2repository/com/testfarm/dependency/1.0	1
varibitkenr/m2repository/com/testfarm/dependency/1.4.5	15
varibitkenr/m2repository/commons-collectors/commons-collectors/3.2.1	1
varibitkenr/m2repository/org/odahu/jackson/jackson-core-asrt/9.12	1
varibitkenr/m2repository/hibernate/hibernate-validator/5.1.1.Final	1

Total: 22

Page générée: 27 Nov. 2017 10:23:17 CET [REST API](#) Jenkins ver. 2.42



Métrique : accessibilité



Métrique : performance (ex : Gatling)



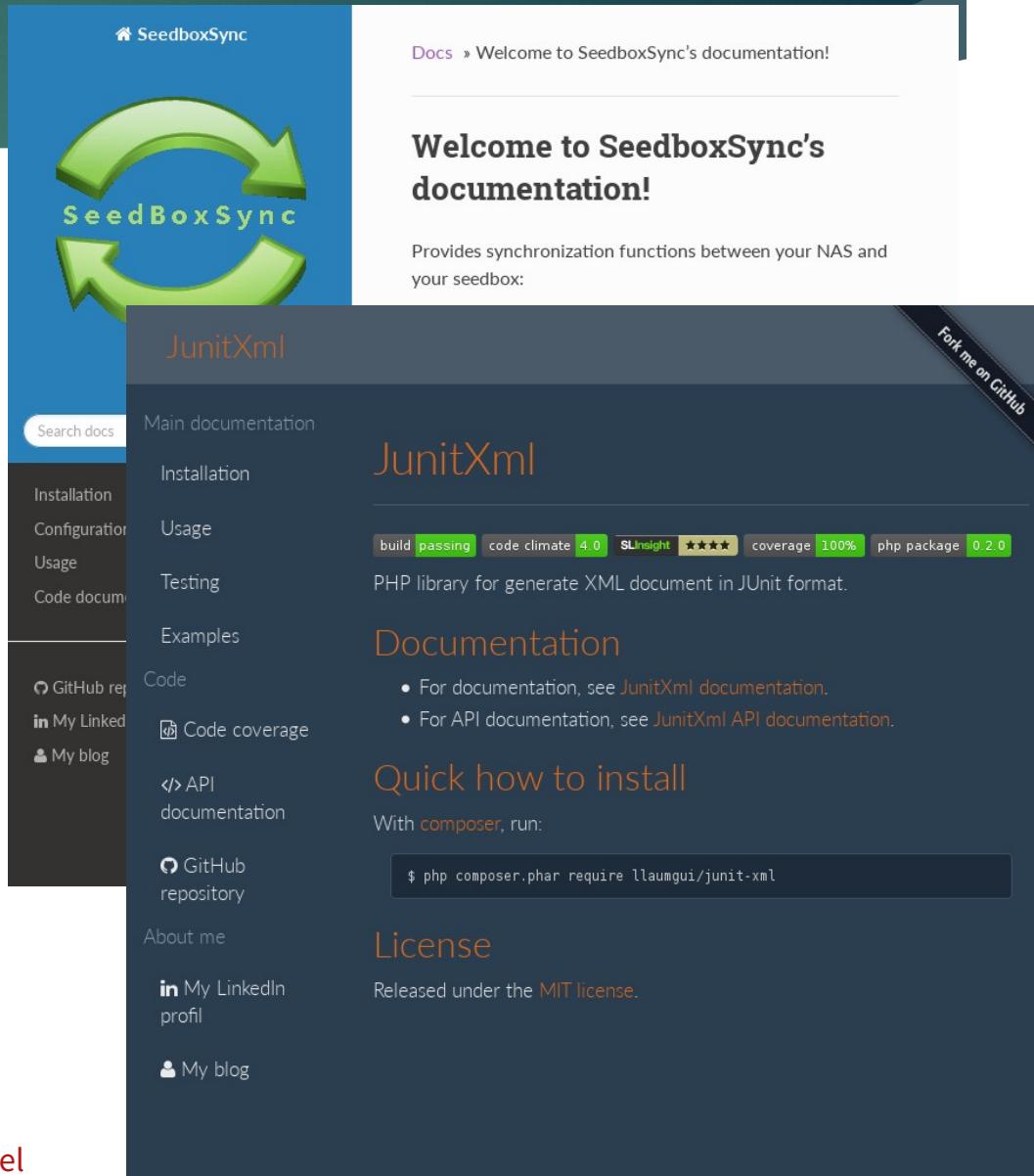
Métrique : divers

- Ligne de code ([phploc](#)).
- Performance Web ([YSlow](#)).
- Bien sûr tous les outils de TU / TI / TNR.

Documentation

Des outils permettent de construire des mini-sites très facilement faisant office de documentation Parmi eux, citons :

- [Couscous](#) (PHP).
- [Sphinx](#) (Python).

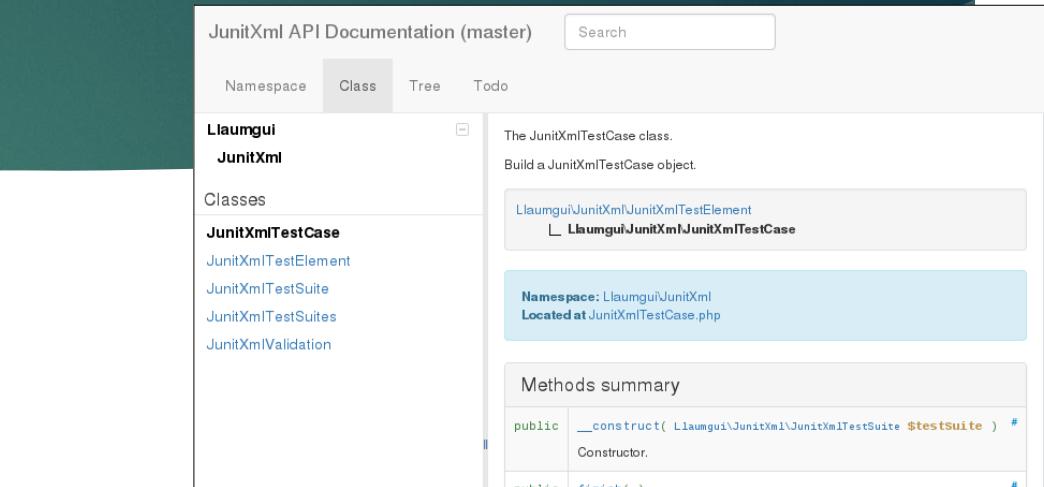


The screenshot shows the documentation page for the **JunitXml** library, which is part of the **SeedboxSync** project. The top navigation bar includes links for [Docs](#), [Welcome to SeedboxSync's documentation!](#), and a [GitHub fork button](#). The main content area features a large logo for **SeedBoxSync** with a green circular arrow icon. Below the logo, the title **JunitXml** is displayed, along with a "Main documentation" section containing links for Installation, Usage, Testing, Examples, Code, and API documentation. A sidebar on the left provides links to the GitHub repository, LinkedIn profile, and blog. The right side of the page contains sections for **Documentation** (with links to documentation and API documentation), **Quick how to install** (instructions for Composer), and **License** (MIT license information). The footer of the page includes build status badges for Travis CI, Code Climate, and SLIInsight.

API documentation

Des outils permettent de construire la documentation de l'API. Parmi eux, citons :

- [apigen](#), [phpDocumentor](#), [Sami](#) (PHP).
- [Sphinx](#) (Python).
- [Javadoc](#) (Java).
- [Doxygen](#) (multiple langages).



The screenshot shows the JUnitXml API Documentation interface. At the top, there's a navigation bar with tabs: Namespace, Class (which is selected), Tree, and Todo. Below the tabs, the namespace Llaumgui is selected, and under it, the JunitXml class is shown. A tooltip provides a brief description: "The JunitXmlTestCase class. Build a JunitXmlTestCase object." Below the class name, a list of methods is provided: JunitXmlTestCase, JunitXmTestElement, JunitXmTestSuite, JunitXmTestSuites, and JunitXmValidation. To the right, a "Methods summary" section shows two methods: __construct and finish. The __construct method is described as a constructor, and the finish method has a parameter \$message. Below this, a "seedboxsync package" section is shown, featuring a logo for SeedboxSync and a green circular arrow icon. It lists sub-modules: seedboxsync.helper module and seedboxsync.transport module. The seedboxsync.helper module is described as a helper module with classes, and its Helper class is detailed, showing its inheritance from object and its static get_torrent_infos method which takes a torrent_path parameter. The method is annotated with a classmethod and log_print message, msg_type=info parameters.

Exemple d'outil de qualimétrie #1 : SensioLabsInsight

SensioLabsNetwork ▾  Symfony 3 Certification is now available in one of our 4,000 exam centers around the world  Guillaume Kulakowski ▾

SensioLabsInsight

Dashboard Help What we analyze Pricing Blog Account

llaumgui / JunitXml 

 Analyzed a day ago by llaumgui, duration: a few seconds
llaumgui / JunitXml #10 (develop)
No alerts on this project. All green!

 Analyzed 7 days ago by llaumgui, duration: a few seconds
llaumgui / JunitXml #9 (develop)
No alerts on this project. All green!

9 days ago
 The grade of **llaumgui / JunitXml** changed from **gold** to **platinum**

 Analyzed 9 days ago by llaumgui, duration: a few seconds
llaumgui / JunitXml #8
No alerts on this project. All green!
 -2

 Analyzed 9 days ago by llaumgui, duration: a few seconds
llaumgui / JunitXml #7

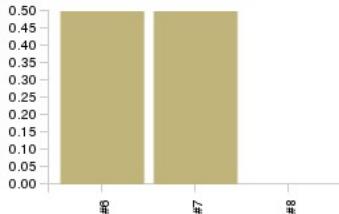
Details

Source: <git@github.com:llaumgui/JunitX...>
Project type: PHP library
Default branch: master

Stats

Lines of code: 282
Remediation cost evolution (in hours)

Only analyses on branch *master* are shown.



Exemple d'outil de qualimétrie #2 : CodeClimate

CodeClimate / llaumgui/JunitXml

Home Features Pricing Sign Up with GitHub Login

Feed Code Issues Branches Trends Tweet 70d137e1

Summary of February 22nd - 28th
27 files changed, 586 insertions, 426 deletions

+ One class/module was added. 6 days ago

src/JunitXmlValidation.php

+ One class/module was added. 9 days ago

tests/PhpUnitHelper.php

Summary of June 1st - 7th
15 files changed, 939 insertions, 312 deletions

Test coverage has improved to 100.0% (+91.5). 9 months ago

+ Four classes/modules were added. 9 months ago

src/JunitXmlTestElement.php

tests/JunitXmlTestCaseTest.php

tests/utils.php

Search by name

4.0 GPA

code climate 4.0 coverage 100%

Link to Code Climate from your README.

100.0% Test Coverage

Hotspots

Huzzah! This repo has no classes or modules worse than a B.

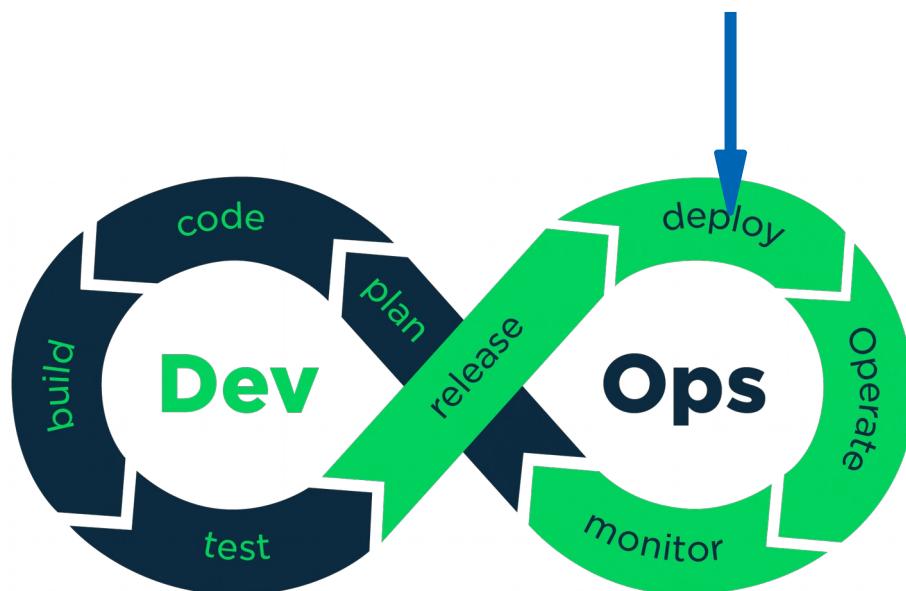
L'expert technique / Lead developer

- Son rôle est primordiale,
- Il doit suivre tous ces indicateurs,
- Il doit faire des revues de code.

5 – Deploy Déploiement Continu

- Définition
- Exemple

Le cycle DevOps



Jenkins



maven



Définition

Le déploiement continu, prolongement de l'intégration continue, est une pratique visant à réduire le plus possible le temps de cycle, le temps écoulé entre l'écriture d'une nouvelle ligne de code et l'utilisation réelle de ce même code par des utilisateurs finaux.

L'équipe s'appuie sur une infrastructure qui automatise l'ensemble des étapes de déploiement (ou "mise en production"), de sorte qu'après chaque intégration qui se solde par des tests passant avec succès, l'application en production est mise à jour (*© institut-agile*).

Discours d'un autre temps...



L'outillage du DevOps

- Sans passer par des concepts de conteneurs (ex : [Docker](#)) qui permettent de pousser une infrastructure complète, des outils comme [Capistrano](#), [Mina](#) ou [Deployer](#) permettent de pousser du code en production.
- Pousse sur **différents serveurs en même temps** (plus de facteur humain).
- Effectue des tâches post-update (ex : vidage de cache *warmup*, *updater*, etc.).
- Autorise le *rollback*.
- Autorise le *staging*.
- Peut-être connecté à une PIC (la boucle est bouclée).

L'outillage du DevOps

Introduction à Docker

Par Guillaume KULAKOWSKI

Expert Technique en Solutions Open Source @ [CGI](#).

Ambassadeur & Packageur @ [FedoraProject](#).

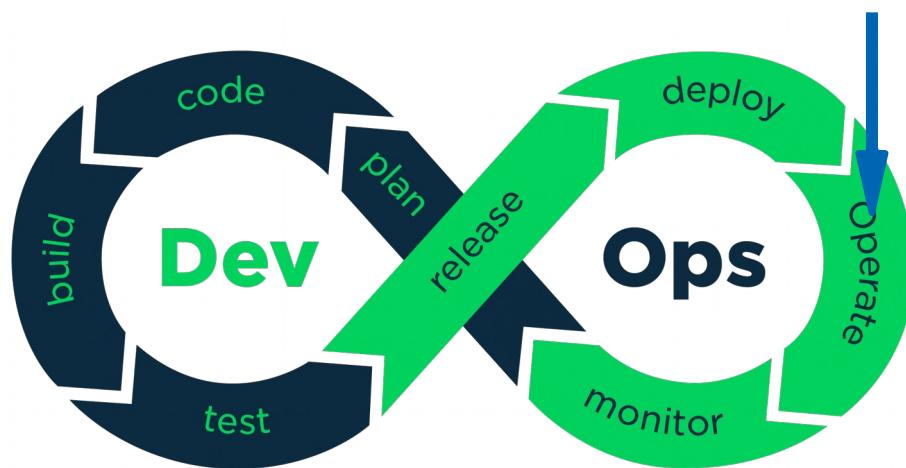
Blogueur @ [blog.kulakowski.fr](#).



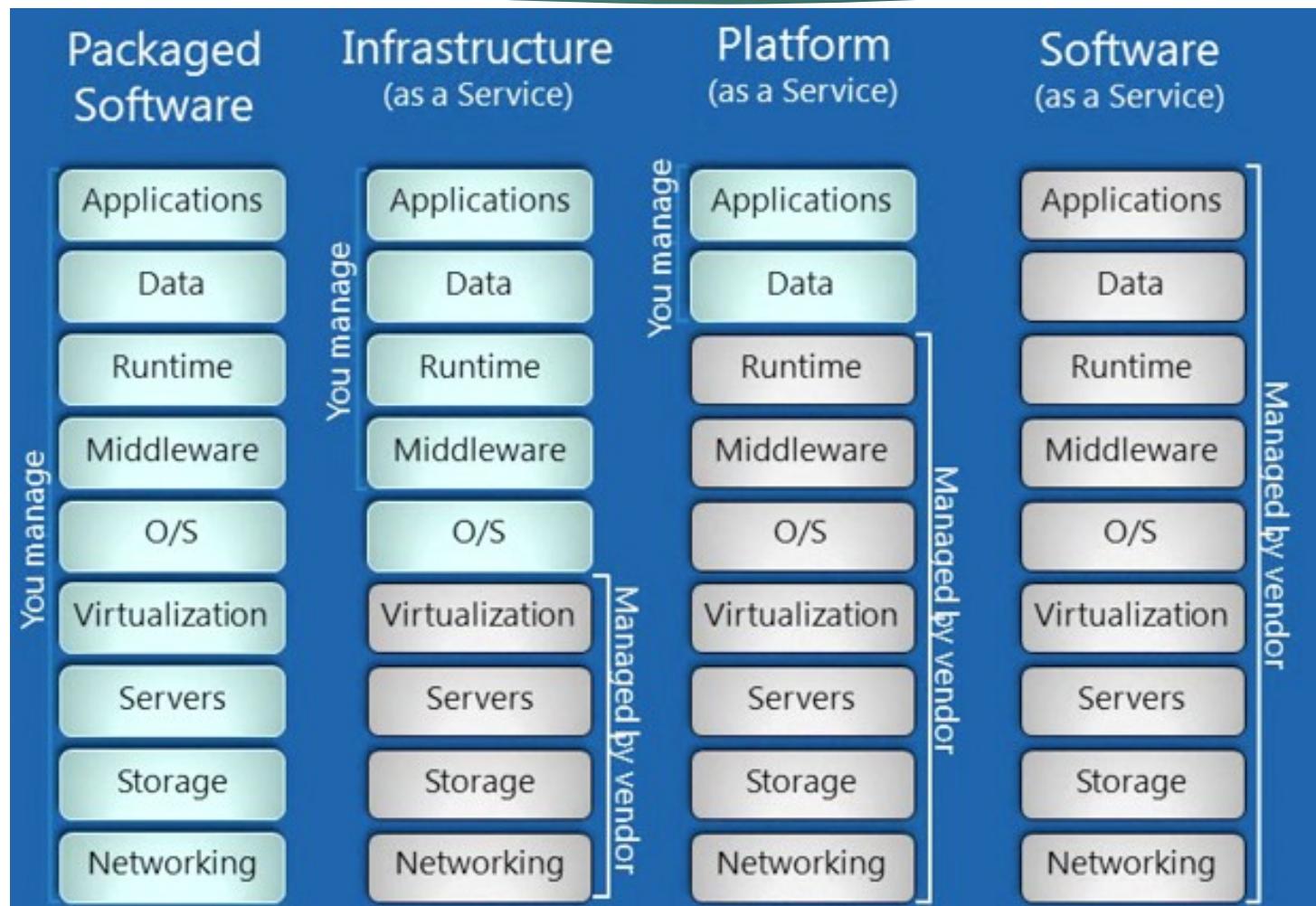
6 – Operate

- Cloud

Le cycle DevOps



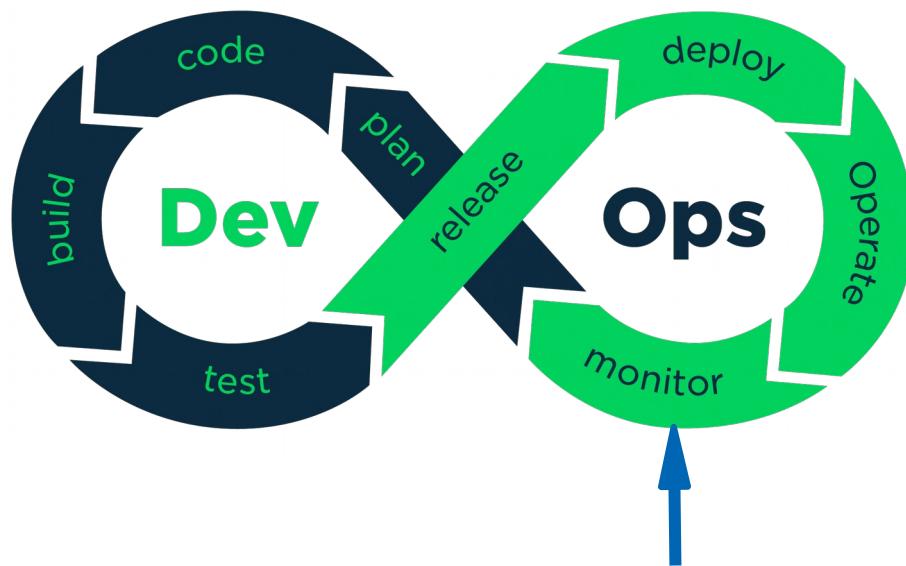
Cloud



7 – Monitor

- Définition
- Exemple

Le cycle DevOps



graylog

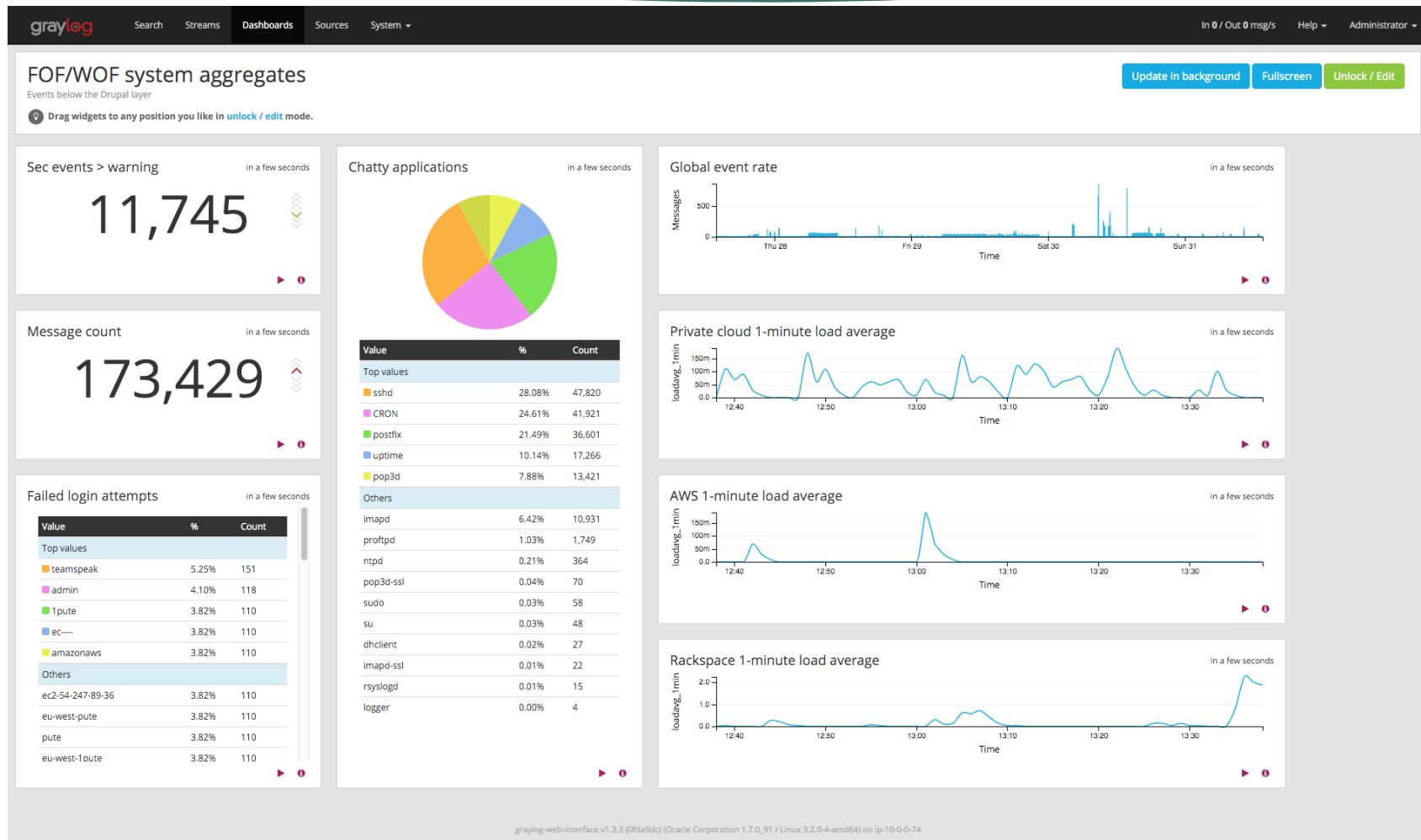
dynatrace

New Relic.
graphite
projects

blackfire

elastic

Exemple de dashboard GrayLog



8 – Outils divers

- Kanban
- Communication

Trello: Kanban agile

MDN Content Team Status ★ Public

On Hold (committed but no work on it for the moment)

- MercrediDocs/WednesdayDocs /MittwochDoks Q2 2015
- Learning Area: 1st PathWay
- Refactor HTMLElement
- Refactor HTML "global attributes" documentation
- Auto-generating API reference docs
- CSS Tutorial
- JS Tutorial
- Review Web Animations API docs

Doing (Task committed and actively worked on this week)

- Q1: Overhaul JavaScript guide
- Follow-up triage meeting 2015-02-17/24 +03-03/10
- Content Kits prototype content
- Follow-up triage meeting 2015-02-10
- Follow-up triage meeting 2015-02-03
- Follow-up triage meeting 2015-01-27
- Web Components - Custom Elements

Review needed

- Document Fetch API
- Channel messaging API
- Service Workers (Review and complete docs)
- Apps quickstart (formerly Recroom)
- Update Web workers
- CSS help in DevTools

No update in the last 14 days. PLEASE UPDATE

- Contribution Pathways

Completed in March 2015

- Write community blog article about MDN curriculum fellowship
- MercrediDocs/WednesdayDocs /MittwochDoks Q1 2015
- MDN 10th Anniversary Plan & brief
- Content Kits meeting with tech evangelism
- Social Media Plan for MDN
- Inheritance Diagrams

Completed in February 2015

- FOSDEM talk
- jFokus Conf Florian Feb 2 - 4
- Web Compat Summit Mt. View.
- Mt. View sprint APIRef / sidebar
- Firefox 36 for developers
- Follow-up triage meeting 2014-11-25
- Glossary: Guidelines for using the glossary template
- Update Learning Area Meta documentation
- Follow-up triage meeting 2014-12-00

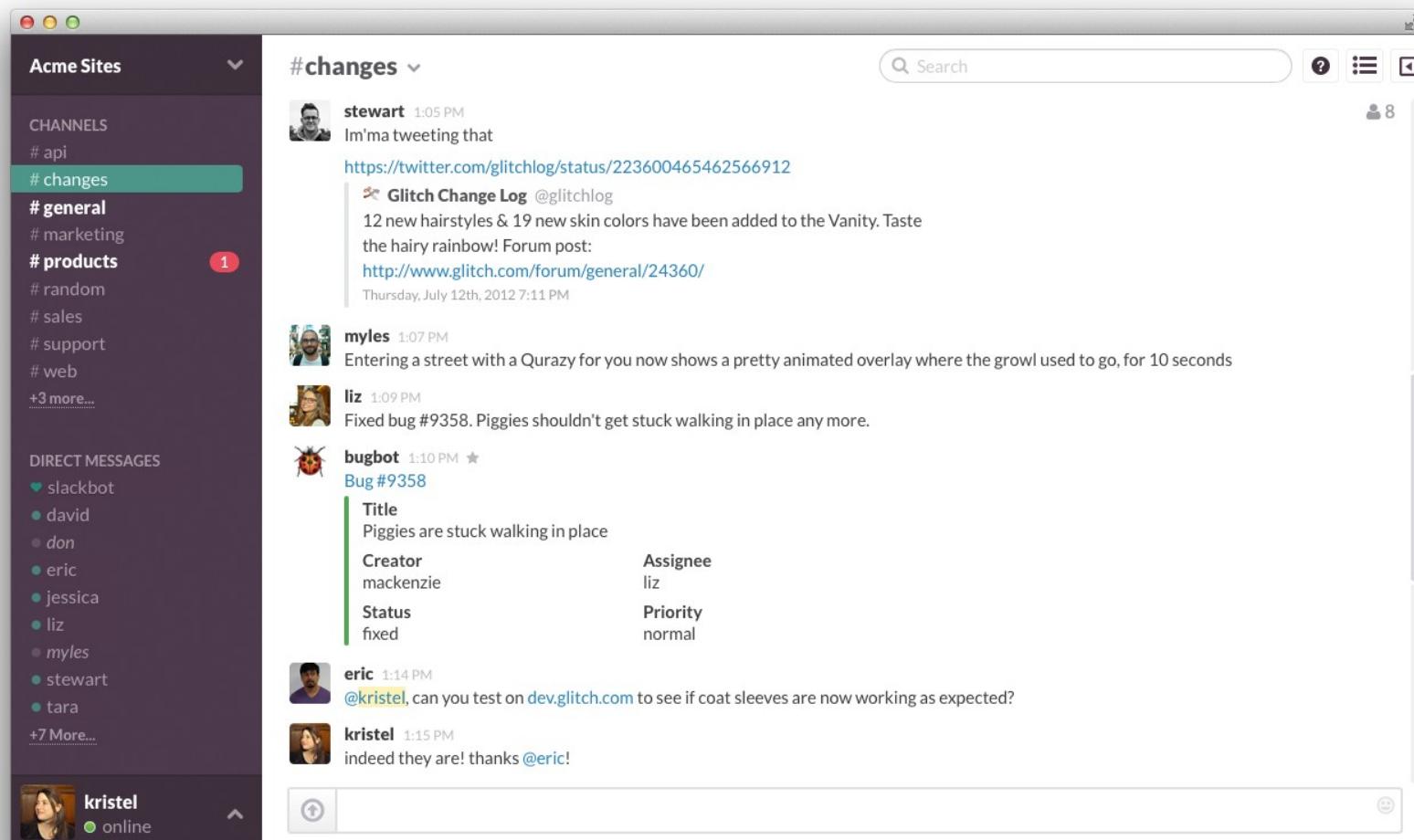
Menu

Members

Activity

- Jeremie Patonnier on MercrediDocs/WednesdayDocs/MittwochDoks Q2 2015
 - added Jean-Yves Perrier
 - added Florian Scholtz
 - added Ali S
 - joined
- Jeremie Patonnier on MercrediDocs/WednesdayDocs/MittwochDoks Q2 2015
 - I need to find a sponsor for May 6th as I will be PTO and not available to host the event.
- Jeremie Patonnier added MercrediDocs/WednesdayDocs/MittwochDoks Q2 2015 to On Hold (committed but no work on it for the moment) and added Q2 Date. Mar 27 at 15:52
- Mark Giffin moved Web Components - Custom Elements from On Hold (committed but no work on it)

Slack : communication entre équipes



Merci à vous !

