



Continuous Build, Integration,
Deployment and Delivery

- I. **CB, CI, CD** : what are those weird acronyms ?
- II. How does it work ?
- III. Why is it useful ?
- IV. Best practices
- V. Common mistakes to avoid
- VI. The competitors
- VII. Feedback from project experience in the banking sector

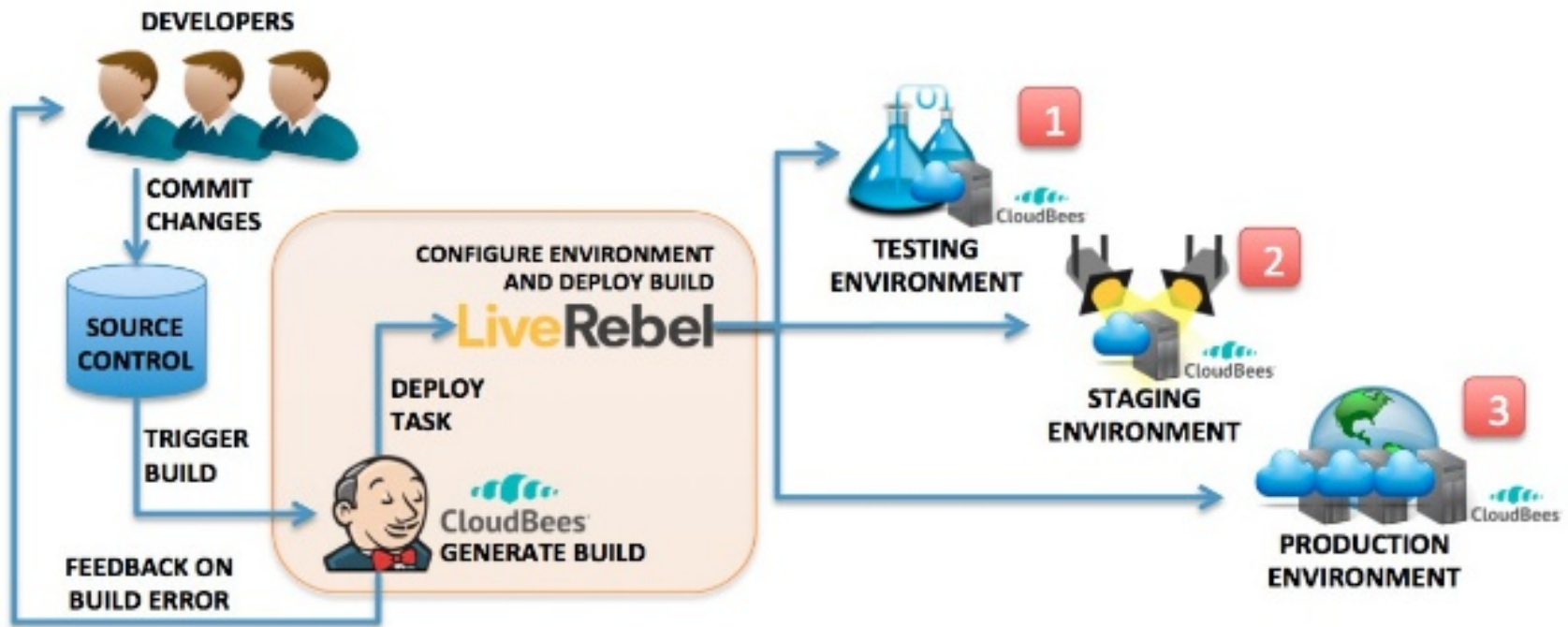
CB, CI, CD : what are those weird acronyms ?

- Continuous Build is based on Unit Tests
- Continuous Integration relates to the code's environment
- Continuous Delivery is for testers and demos makers
- Continuous Deployment is for infrastructure purposes

How does it work ?



Example Production CI Workflow



How does it work ?

1. Developers work on features and run the Unit Tests locally
2. Changes are pushed to the VCS ; application is built and Integration tests are run (triggered, scheduled or on-demand)
3. Upon demand, the Deployment Pipeline may be triggered (can involve code quality and code security scans)
4. If all the lights are green, the build is deployed on Test/QA environments

Why is Continuous Integration useful ?

- A successful build does not mean a working build
- **Fail-fast** : developers know immediately if their changes broke something
- Smaller and more frequent changes mean easier debugging and **faster fixing**
- Every change is tested against the **entire environment**
- Avoid defects survival until deployment
- **Visibility** is granted for the whole team
- The whole process is **automated** (build servers)
- Part of Xtreme programming, Test-driven development, DevOps and **Agile** methodologies

Why are Continuous Deployment and Delivery useful ?



- Every new build is tested against the **entire environment and infrastructure**
- **Visibility** is granted for the whole team
- The whole process is also **automated**
- Services can be **virtualized**
- Stable builds are always available for testing and demo purposes
- Deployment steps can be measured in order to anticipate production issues (metrics gathering)

- Developers check in their working and tested changes every day
- Developers check out every morning
- Make the Test/QA environment a **clone of the Production**
- **Document** the deployment pipelines (what should happen next ?)
- Use an external provider to host your build and deployment infrastructure if you can
- If you host your own infrastructure, **scale it !**
- Use a central binary repository (e.g Nexus) to store artefacts
- Automate, automate, automate (just like pizza)

Common mistakes



- Avoid best practices for best practices
- A mess is a mess ; continuously integrating a mess won't make it any better
- Build and Deployment infrastructure must be **scaled** and **stress-tested**
- Take your time, analyze deeply, make sure everyone gets the point
- Is it really relevant ?

The competitors



Jenkins



Bamboo



TeamCity



Microsoft®
Visual Studio®
Team Foundation Server

The competitors - Jenkins

Jenkins

[Jenkins](#) » [Gate](#)

New Job

People

Build History

Edit View

Delete View

Project Relationship

Check File Fingerprint

Manage Jenkins

Query and Trigger Gerrit Patches

My Views

Job Config History

Build Queue

No builds in the queue.

Build Executor Status

#	Master
1	Idle
2	Idle
build (offline)	
build1	
1	Idle
build2 (offline)	
build3 (offline)	
build4 (offline)	
ci	
1	Idle
citrix	
1	Idle

All

Burrow

Dashboard

Gate

Glance

Keystone

Milestone-proposed

Nova

OpenStack-CI

Openstack-manuals

Overview

Quantum

Swift

Websites

+

S

W

Name ↓

Last Success

Last Failure

Last Duration

		glance	15 hr (#89283)	5 days 15 hr (#89277)	2 min 3 sec	
		glance-merge	15 hr (#116)	1 mo 2 days (#78)	3 sec	
		glance-pep8	15 hr (#289)	16 days (#269)	6.5 sec	
		keystone	12 hr (#400)	13 hr (#399)	1 min 45 sec	
		keystone-merge	12 hr (#292)	6 days 15 hr (#267)	4.6 sec	
		keystone-pep8	12 hr (#335)	6 days 15 hr (#310)	9 sec	
		keystone-pylint	12 hr (#408)	6 days 15 hr (#383)	22 sec	
		nova	3 hr 44 min (#121729)	19 hr (#121717)	8 min 19 sec	
		nova-merge	3 hr 44 min (#215)	21 hr (#194)	1 min 0 sec	
		nova-pep8	3 hr 44 min (#1588)	21 hr (#1567)	1 min 9 sec	
		quantum	9 hr 37 min (#62)	9 days 9 hr (#56)	9.1 sec	
		quantum-pep8	9 hr 37 min (#36)	N/A	4.4 sec	
		quantum-pylint	9 hr 35 min (#31)	N/A	20 sec	
		swift	21 hr (#119342)	1 mo 17 days (#119327)	14 sec	
		swift-merge	21 hr (#16)	N/A	3.3 sec	

ENABLE AUTO REFRESH

- Infrastructure availability is critical, especially in large corporations
- The earlier you begin, the easier it gets
- Support team must be trained ; external support is a requirement when the business depends on the CI platform
- Always backup, and make it redundant if possible
- Streamline the deployment process for your users



Any Questions ?

info@movify.be