CLOUDOPEN

@

THE LINUX FOUNDATION
OPEN SOURCE SUMMIT
LATIN AMERICA

From Traditional to GitOps

A tale of modernization

#ossummit @tuxtor

THE LINUX FOUNDATION

Cloud native

The false assumption

By migrating to newer cloud technologies -e.g. Serverless, BaaS, Microservices- companies will automatically achieve scale -i.e. support more users with less money - and launch new faster.

Cloud native

Approach for building modern computing systems on dynamic environments such as private and public clouds.

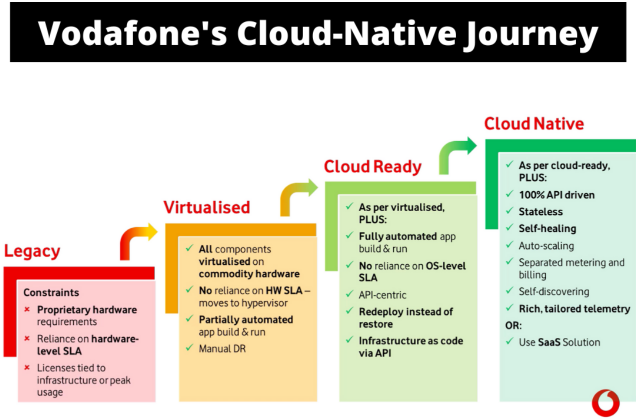
- Reactive systems
- 12 cloud native factors
- Cloud native design patterns
- Domain Driven Design
- Microservices chassis and/or service mesh
- Container orchestration, serverless, BaaS
- Cloud

Cloud native

- (We want) Reactive systems
- (A battle tested approach is) 12 cloud native factors
- (Common mistakes and solutions are described by) Cloud native design patterns
- (Hence we divide the system with) Domain Driven Design
- (And develop the systems by using) Microservices chassis and/or service mesh
- (Running the workloads over) Container orchestration, serverless, BaaS
- (Hence the) Cloud

The Cloud Native migration projects, are in real life Macro-Projects that should be carried out properly

Cloud native



Vodafone's Cloud-Native Journey

Legacy

Constraints

- ✗ **Proprietary hardware** requirements
- ✗ Reliance on **hardware-level SLA**
- ✗ Licenses tied to infrastructure or peak usage

Virtualised

- ✓ **All** components **virtualised** on **commodity hardware**
- ✓ **No** reliance on **HW SLA** – moves to hypervisor
- ✓ **Partially automated** app build & run
- ✓ Manual DR

Cloud Ready

- ✓ **As per virtualised, PLUS:**
- ✓ **Fully automated** app build & run
- ✓ **No** reliance on **OS-level SLA**
- ✓ API-centric
- ✓ **Redeploy instead of restore**
- ✓ **Infrastructure as code via API**

Cloud Native

- ✓ **As per cloud-ready, PLUS:**
- ✓ **100% API driven**
- ✓ **Stateless**
- ✓ **Self-healing**
- ✓ Auto-scaling
- ✓ Separated metering and billing
- ✓ Self-discovering
- ✓ **Rich, tailored telemetry**

OR:

- ✓ Use **SaaS** Solution

Image Credits: Joanna Newman, Vodafone Keynote

The migration project(s)

Cloud native - The government wants to go Cloud Native

Under NDA, but still . . .

- Government institution = Local data required by law
- Institution 1:
  - NodeJS based solution
  - Docker based monolith
  - CI/CD with GitLab
- Institution 2:
  - Java EE based solution
  - Apache TomEE based monolith
  - No CI/CD at all

Both systems provide services for 16m potential users, 1000-5000 concurrent users at any time.
These **units** where part of the same **government sector**.

Cloud native - The PMI approach

- Conception & initiation
  - Software architecture and developer diagnose
- Definition and planning
  - Roadmap definition
- Launch or execution
  - Implementation
  - Acquisitions
  - Training
  - Software development
- Performance & control
  - Tech: Deliverable and quality metrics
  - Project: Budget, deadlines, viability
- Project close
  - Live documentation
  - Continuous improvement

Conception & initiation

No more than two brainstorming sessions. Two hours on average

Mandatory stakeholders:

- Software architect (Tech Lead, Dev. Sr.)
- Infrastructure (Sysadmin, SRE)
- Direct contact point

Key questions:

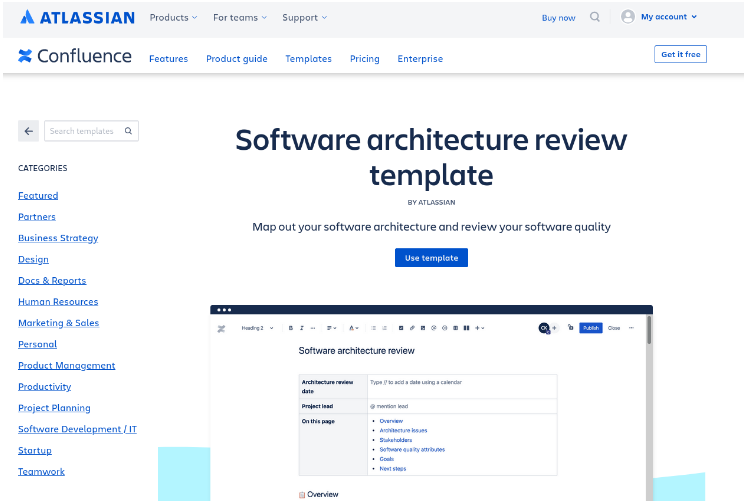
- Motivation
- Current team’s size and skills
- Tech Stacks

Definition and planning

Architecture review

- Issues
- Possible solutions, approaches and **actions**
- Roadmap with options (consultants, outsourcing, training)
- Contracts based on deliverables

Definition and planning



CATEGORIES

- Featured
- Partners
- Business Strategy
- Design
- Docs & Reports
- Human Resources
- Marketing & Sales
- Personal
- Product Management
- Productivity
- Project Planning
- Software Development / IT
- Startup
- Teamwork

Software architecture review template

BY ATLASSIAN

Map out your software architecture and review your software quality

Use template

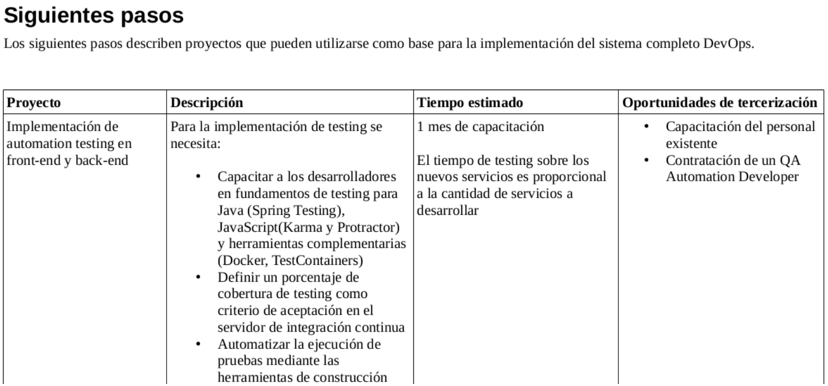
Software architecture review

Architecture review date	Type // to add a date using a calendar
Project lead	@ mention lead
On this page	• Overview • Architecture issues • Stakeholders • Software quality attributes • Goals • Next steps

Overview

Definition and planning

Project - Description - Estimated time - Provided opportunities



Siguientes pasos

Los siguientes pasos describen proyectos que pueden utilizarse como base para la implementación del sistema completo DevOps.

Proyecto	**Descripción**	**Tiempo estimado**	**Oportunidades de tercerización**
Implementación de automation testing en front-end y back-end	Para la implementación de testing se necesita: • Capacitar a los desarrolladores en fundamentos de testing para Java (Spring Testing), JavaScript(Karma y Protractor) y herramientas complementarias (Docker, TestContainers) • Definir un porcentaje de cobertura de testing como criterio de aceptación en el servidor de integración continua • Automatizar la ejecución de pruebas mediante las herramientas de construcción	1 mes de capacitación El tiempo de testing sobre los nuevos servicios es proporcional a la cantidad de servicios a desarrollar	• Capacitación del personal existente • Contratación de un QA Automation Developer

Launch or execution

1-Contact point, 2-Direct means of communication , 3- Non-repudiable means of communication

- Culture
  - DevOps
  - Test Driven Development
- Infrastructure
  - Remote access
  - VCS, CI/CD
  - Cloud Native Platform -e.g. Openshift, Kubernetes, Amazon EKS, Oracle Kubernetes Engine-
  - Observability -e.g. Linkerd, Prometheus, Grafana, ELK, alarms-

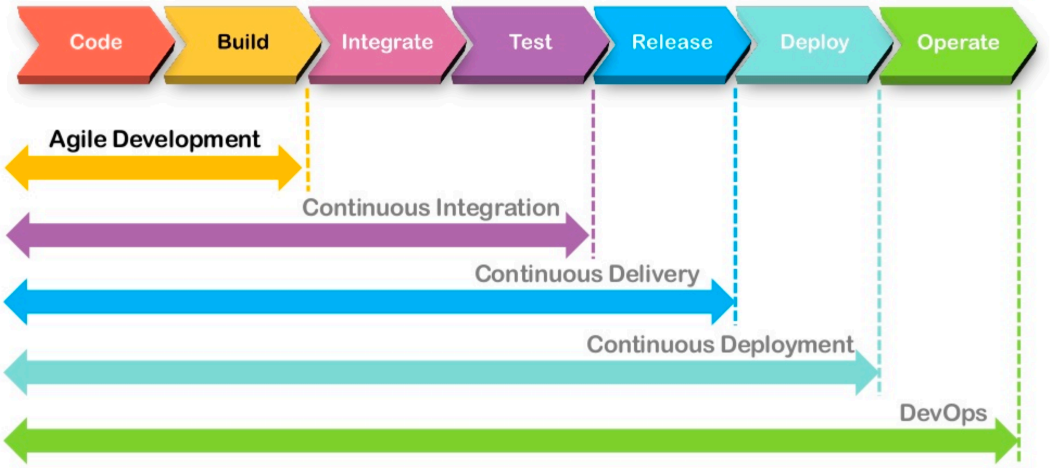
Ideally make one presentation/technology transfer per deliverable.

Launch or execution

All phases produce **”live documentation”**.

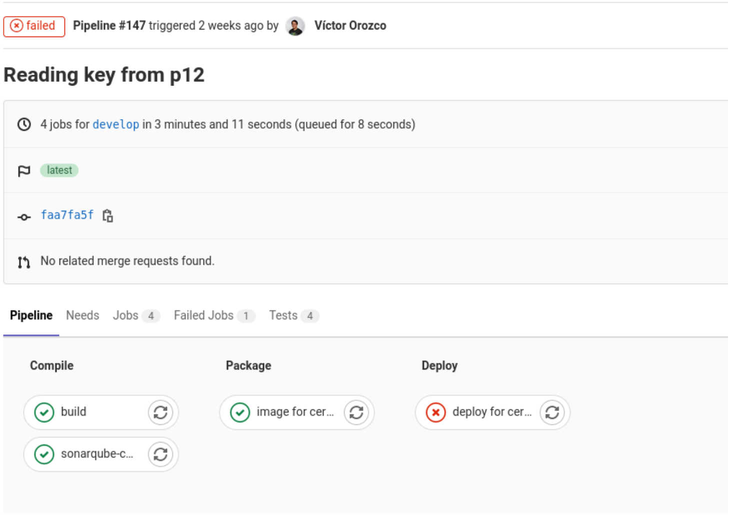
- Training and development
  - Bootstrap archetypes (also pet projects)
  - SCM -e.g. Maven, NPM -
  - TDD, DDD, Microservice Chassis, infrastructure as code
  - CI/CD Pipelines
  - Don’t kill the monolith, create an ecosystems around
  - New project with mandatory cloud native factors

Launch or execution - DevOps



THE LINUX FOUNDATION

Launch or execution - CI/CD



failed **Pipeline #147** triggered 2 weeks ago by **Víctor Orozco**

Reading key from p12

4 jobs for `develop` in 3 minutes and 11 seconds (queued for 8 seconds)

latest

`faa7fa5f`

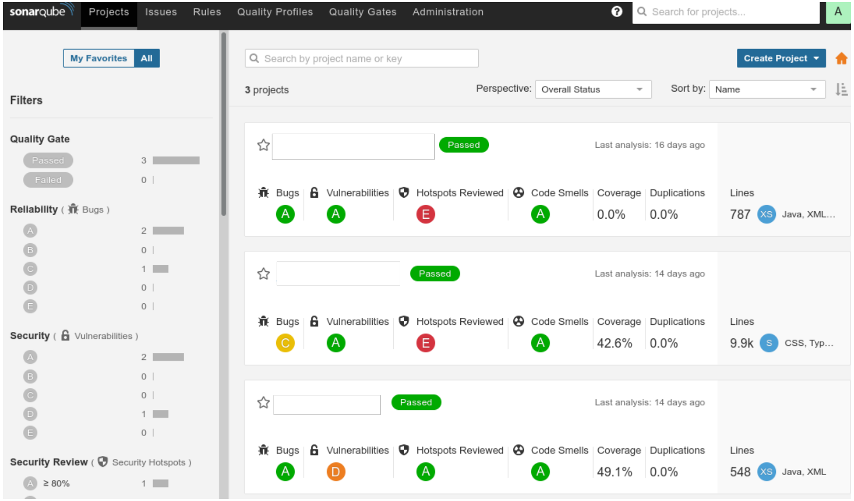
No related merge requests found.

Performance & control

- Tech
  - Code quality-e.g. coverage, code smells, bugs, vulnerabilities-
  - Performance -e.g. network latency and failures-
  - Instrumentation
  - How many services have been migrated
- Project
  - Budget
  - Deliverables vs. deadlines
  - Users and developers perceptions

THE LINUX FOUNDATION

Performance & control - Quality



3 projects

Perspective: Overall Status — Sort by: Name

Filters

Quality Gate

Passed	3
Failed	0

Reliability (Bugs)

A	2
B	0
C	1
D	0
E	0

Security (Vulnerabilities)

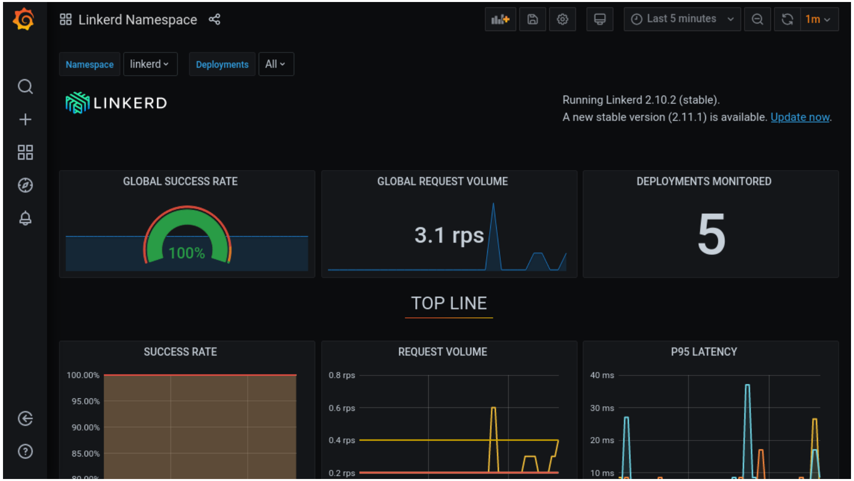
A	2
B	0
C	0
D	1
E	0

Security Review (Security Hotspots)

A ≥ 80%	1

Quality Gate	Last analysis	Bugs	Vulnerabilities	Hotspots Reviewed	Code Smells	Coverage	Duplications	Lines	Size	Languages
Passed	16 days ago	A	A	E	A	0.0%	0.0%	787	XS	Java, XML...
Passed	14 days ago	C	A	E	A	42.6%	0.0%	9.9k	S	CSS, Typ...
Passed	14 days ago	A	D	A	A	49.1%	0.0%	548	XS	Java, XML

Performance & control - Instrumentation

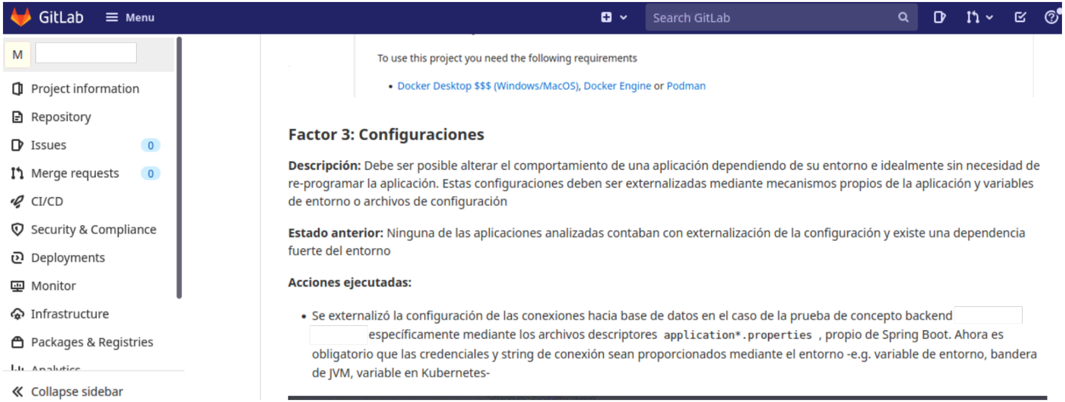


Running Linkerd 2.10.2 (stable).
A new stable version (2.11.1) is available. Update now.

Project close

- Implementation **should** transition to support
- Live documentation
- What could be done better?

Project close - Live documentation



To use this project you need the following requirements

- Docker Desktop $$$ (Windows/MacOS), Docker Engine or Podman

Factor 3: Configuraciones

Descripción: Debe ser posible alterar el comportamiento de una aplicación dependiendo de su entorno e idealmente sin necesidad de re-programar la aplicación. Estas configuraciones deben ser externalizadas mediante mecanismos propios de la aplicación y variables de entorno o archivos de configuración

Estado anterior: Ninguna de las aplicaciones analizadas contaban con externalización de la configuración y existe una dependencia fuerte del entorno

Acciones ejecutadas:

- Se externalizó la configuración de las conexiones hacia base de datos en el caso de la prueba de concepto backend [redacted] [redacted] específicamente mediante los archivos descriptores `application*.properties` , propio de Spring Boot. Ahora es obligatorio que las credenciales y string de conexión sean proporcionados mediante el entorno -e.g. variable de entorno, bandera de JVM, variable en Kubernetes-

Víctor Orozco

Duke's Choice Award
2016 Winner
Java

Oracle Groundbreakers

Java Champions

ORACLE Certified Professional Java SE 8 Programmer

ORACLE Certified Associate Java SE 8 Programmer

- vorozco@nabenik.com
- @tuxtor
- https://vorozco.com
- https://tuxtor.shekalug.org

CLOUDOPEN

@

THE LINUX FOUNDATION
OPEN SOURCE SUMMIT
LATIN AMERICA

THE LINUX FOUNDATION