



Ingeniería Web: Visión General

-IWVG-

DevOps

Jesús Bernal Bermúdez

¿Qué es DevOps?

Development – Operations & QA

Prácticas para unificar el desarrollo de software y la operación de negocio

El objetivo es integrar los procesos y las herramientas para optimizar la entrega continua de software de calidad, siguiendo los objetivos empresariales

DevOps

Prácticas

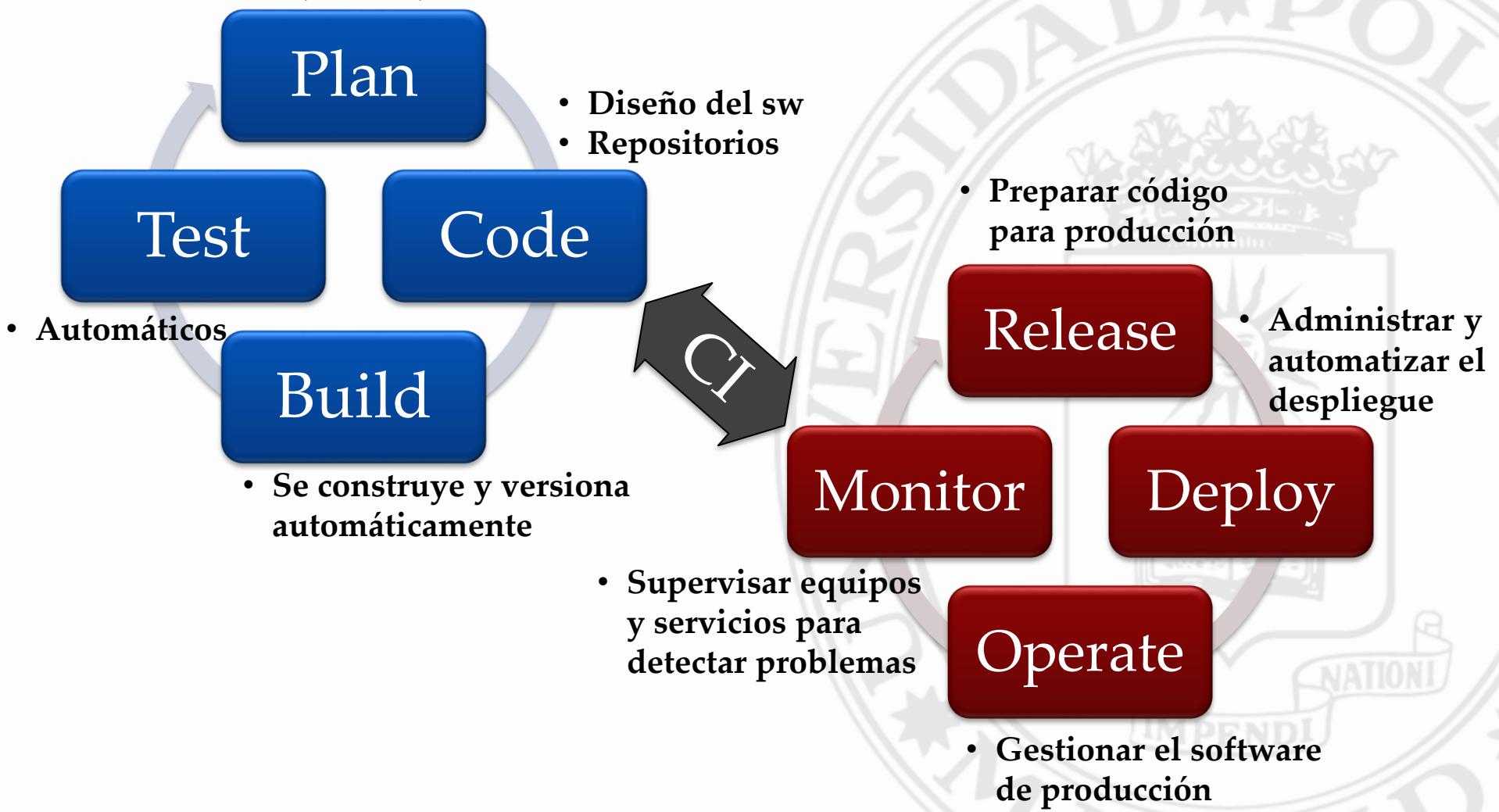
- Automatización de todos los procesos:
 - Integración Continua (CI).
 - Entrega/Despliegue Continuo (CD).
- Ciclos de desarrollo cortos.
- Microservicios.
- Lanzamientos confiables.
- Sistema seguro, estable y rápido.
- Sistemas monitorizados.
- Comunicación y colaboración.

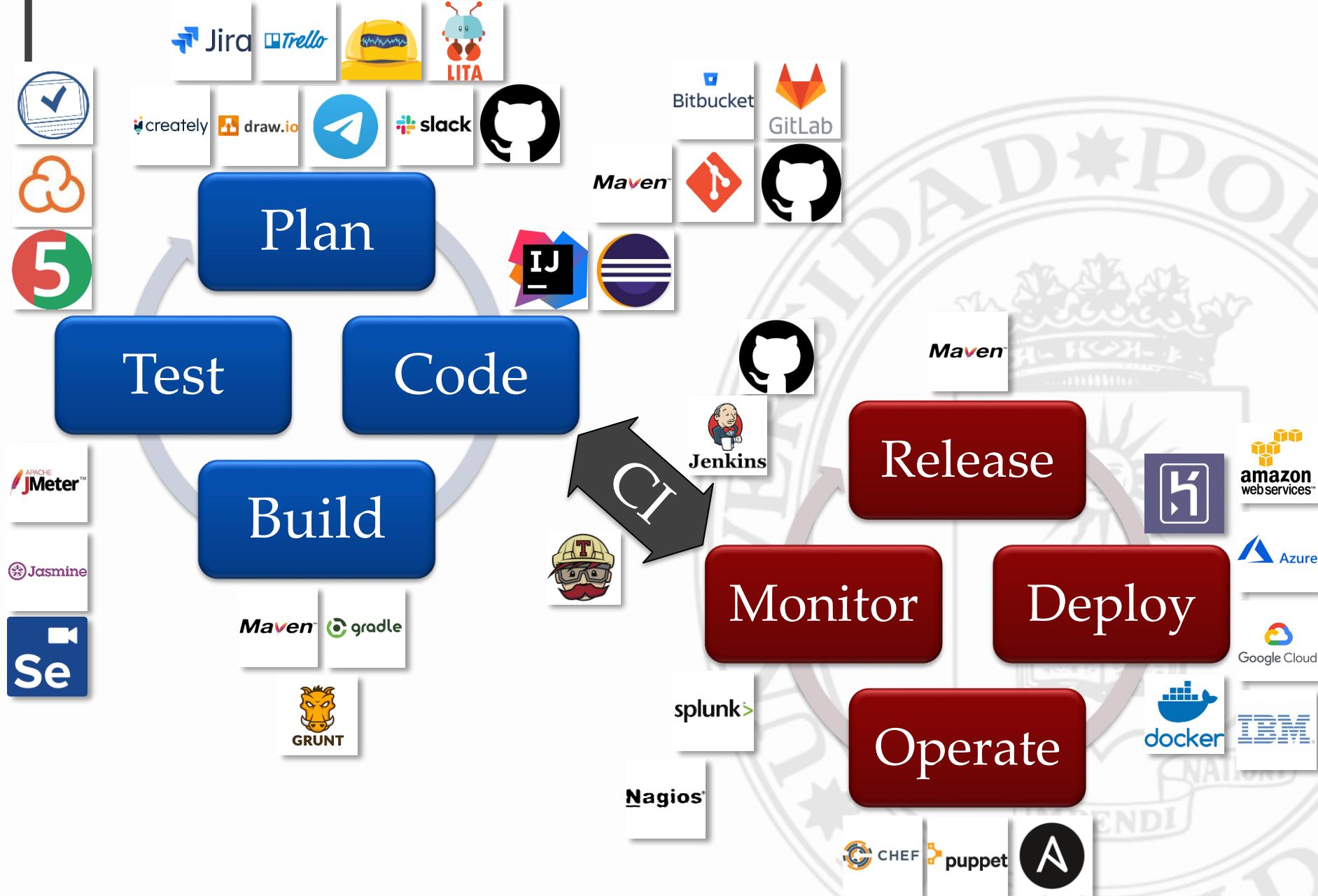
¿Qué es un Ecosistema Software?

Ecosistema Software

Se define como un espacio de trabajo donde un conjunto de herramientas interactúan y funcionan como una unidad para la colaboración, desarrollo, despliegue y supervisión del software en todas sus fases

- Valor comercial
- Gestión del proyecto
- XP, Scrum, Kanban...





Programación Orientada a Objetos

Programación Funcional

POO

Programación imperativa:
¿Cómo?.

Abstracción, encapsulamiento,
modularidad, jerarquía.

Clases relacionadas mediante
herencia, composición, agragación
y asociación.

Objetos con estado: los atributos.

**Unidad: Clases &
Objetos.**

PF

Programación declarativa: ¿Qué?.
Basado en Funciones: funciones
Lambda.

Sin estado, sin orden y sin efectos
colaterales.

Valores inmutables: paso de
parámetros por valor.

Recomendado *Optional* frente a
Exception

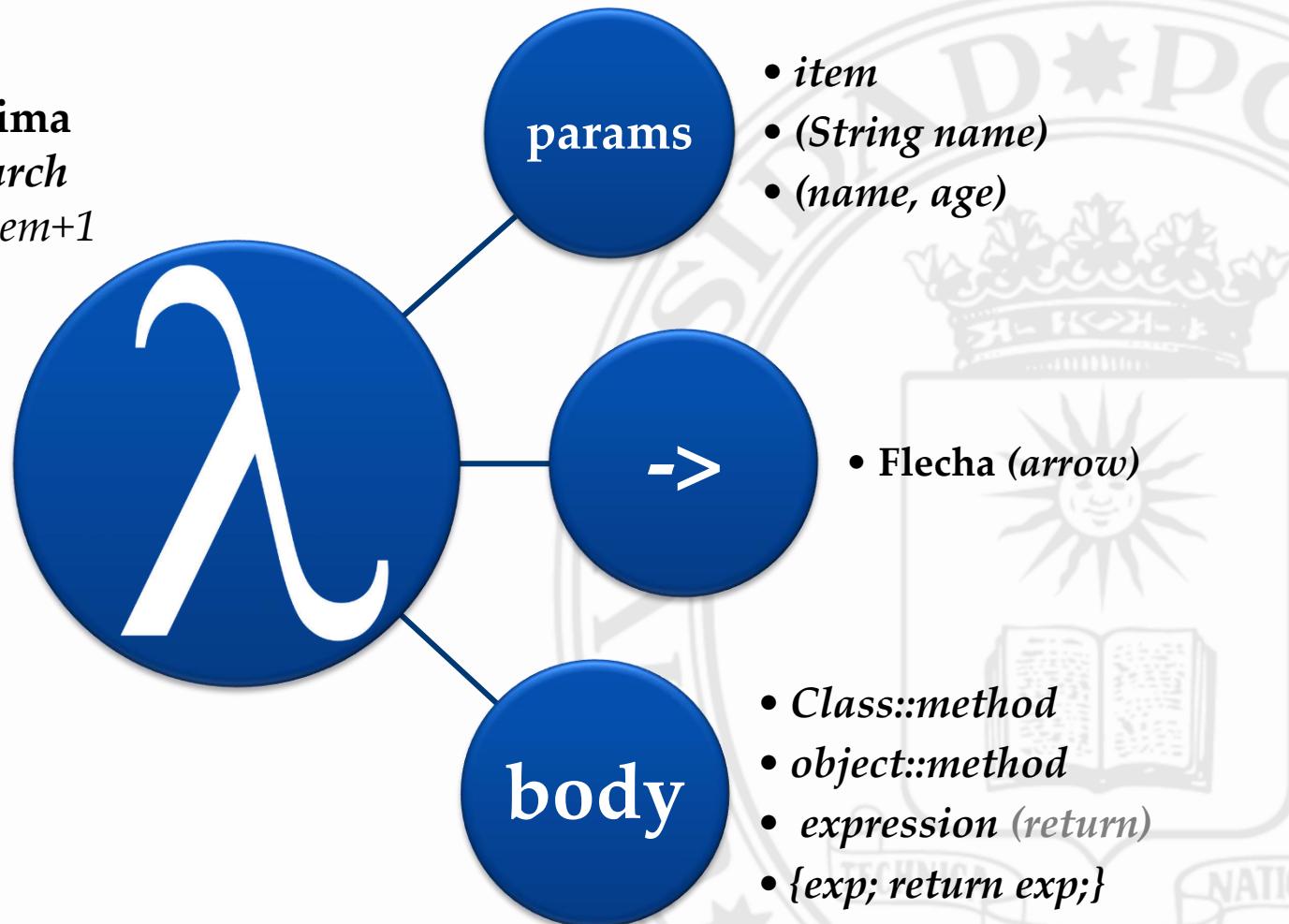
Unidad: Función.

Java

Función Lambda

Función anónima

- *Alonzo Church*
- Ej. *item -> item+1*



Java

Función Lambda

Consumer<T> accept(T)

- System.out::println

Function<T,R> apply(T):R

- item -> item +1

Predicate<T> test(T):boolean

- item -> item > 0

Supplier<T> get(): T

- ()-> "..."

BiConsumer<T,U,R> apply(T,U):R

- (msg1, msg2) -> System.out.println(msg1 + "," + msg2)

BiFunction<T,U,R> accept(T,U):R

- (x, y) -> x + y

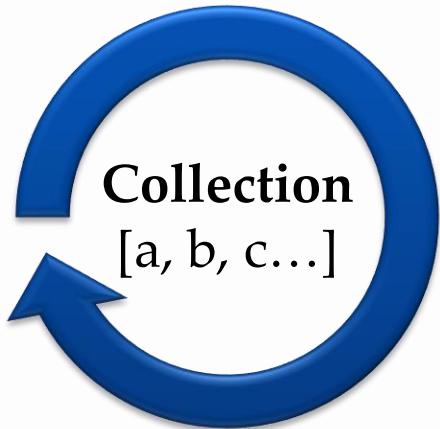
BiPredicate<T,U,R> apply(T,U):R

- String::equals

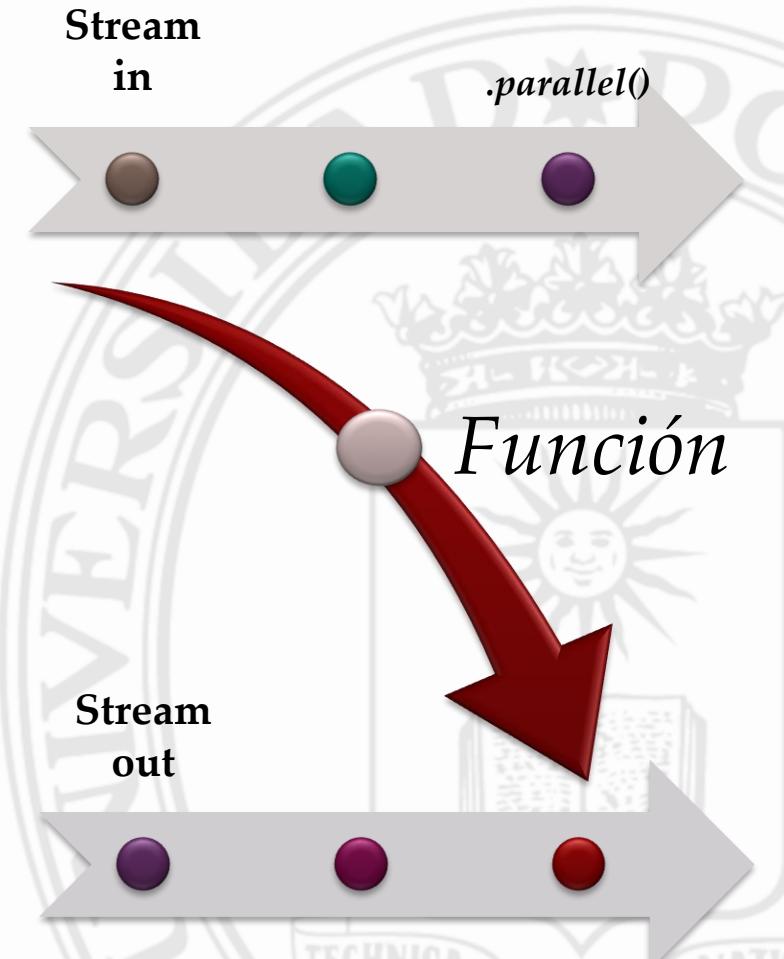
...

Java

Collection & Stream



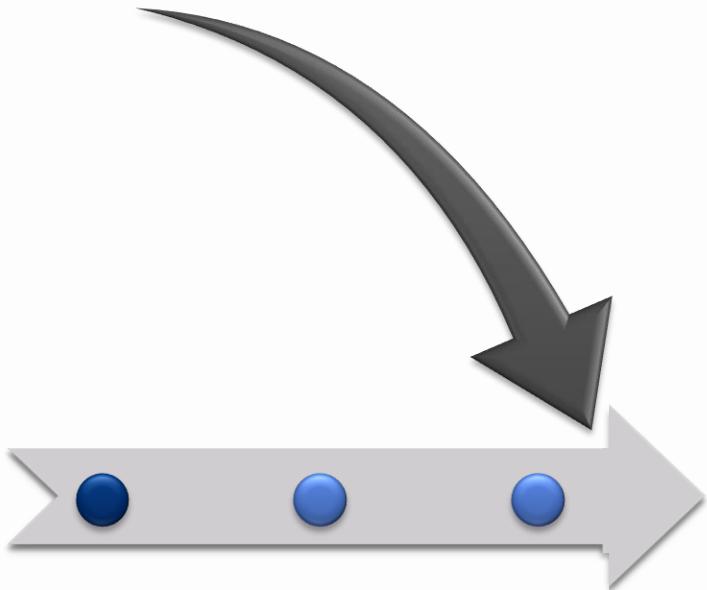
- Gestión bucle
- *Método de transformación*
- Paralelismo
- ¿Reutilización del código?



Java

Stream. Crear y colección

```
list.stream();  
Stream.range();  
Stream.of(...);  
Stream.generate();  
Stream.iterate();
```

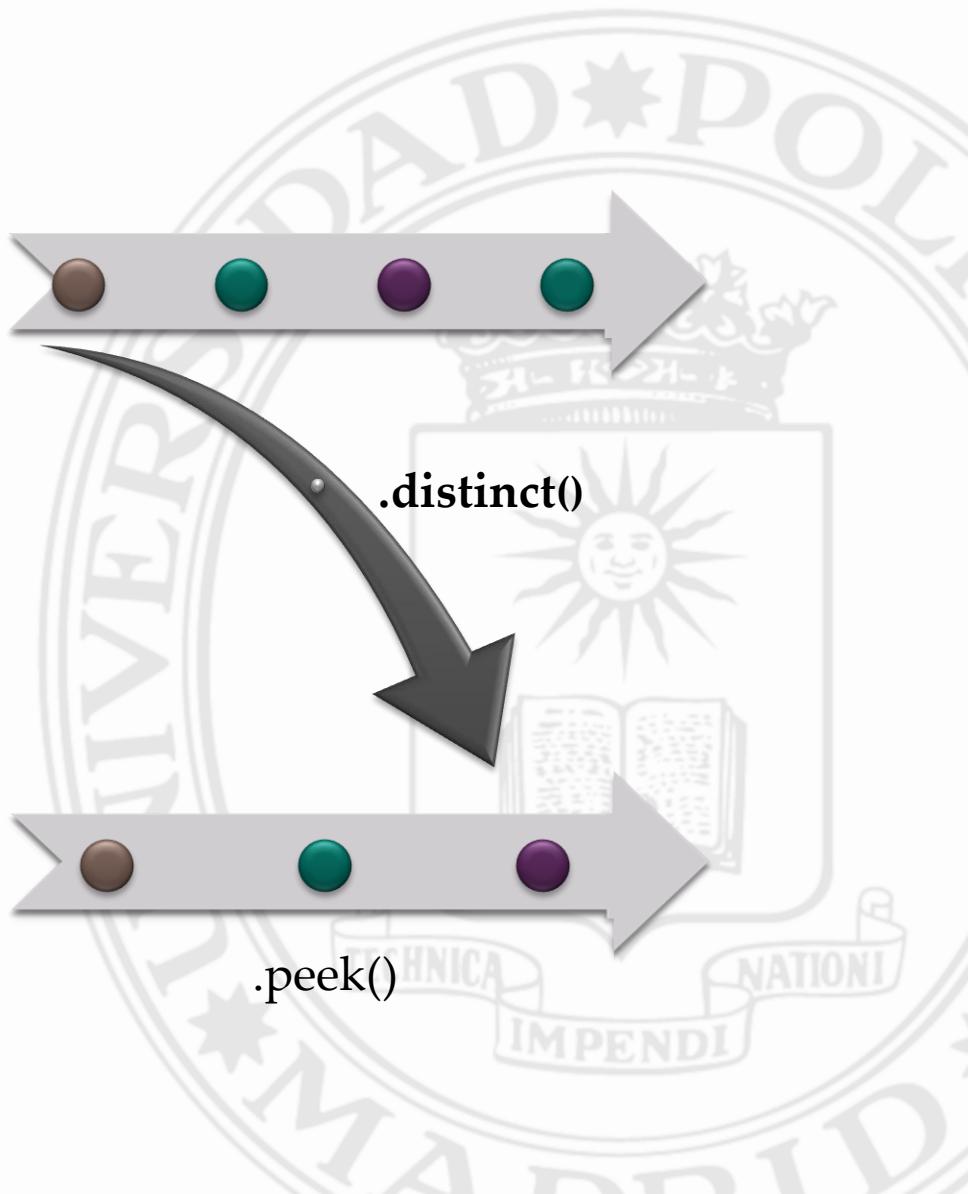
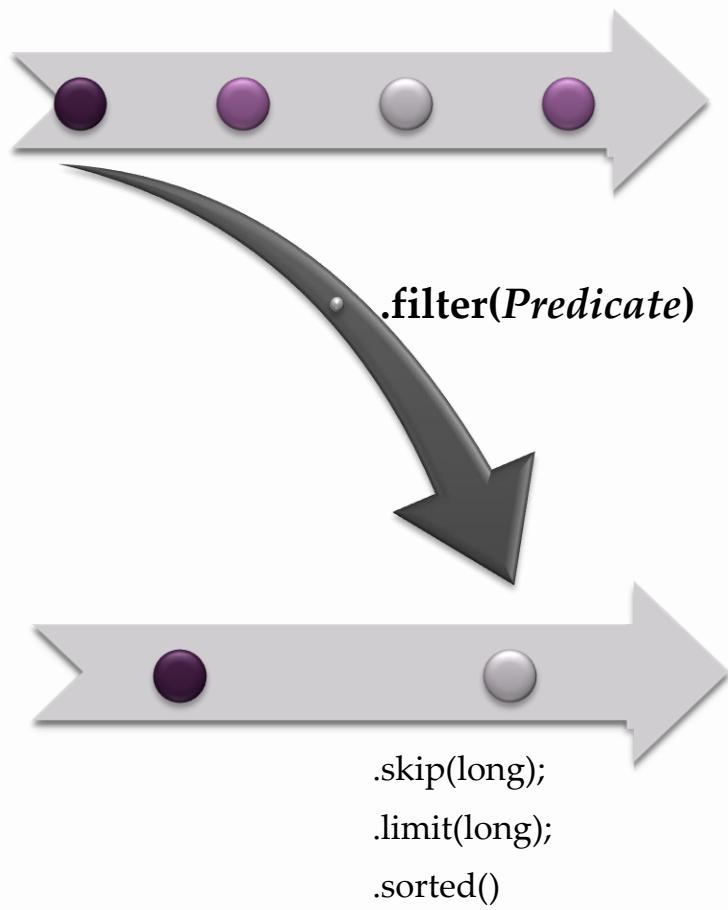


```
.collect(Collectors.toList())  
.toArray(Integer::new)
```

[a, b, c]

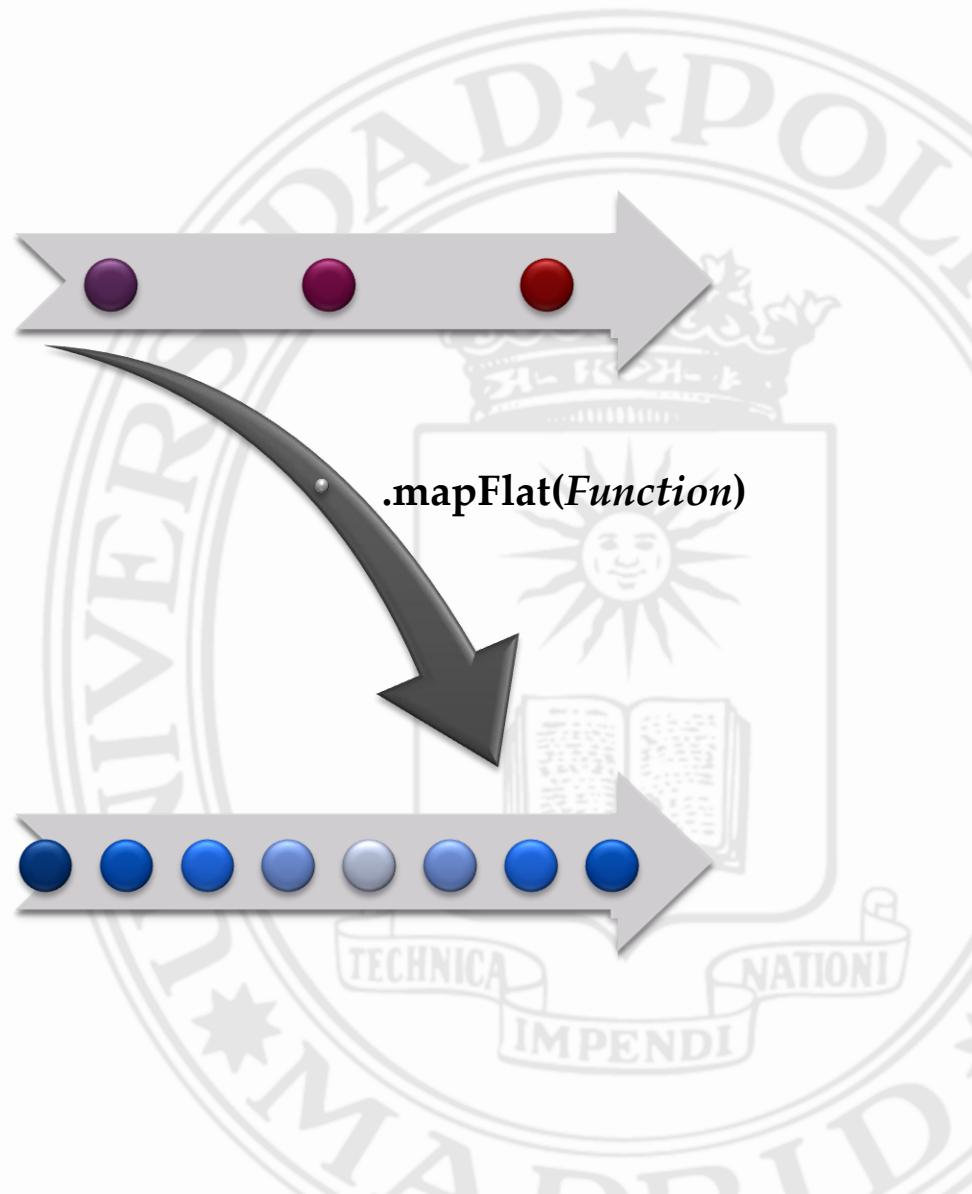
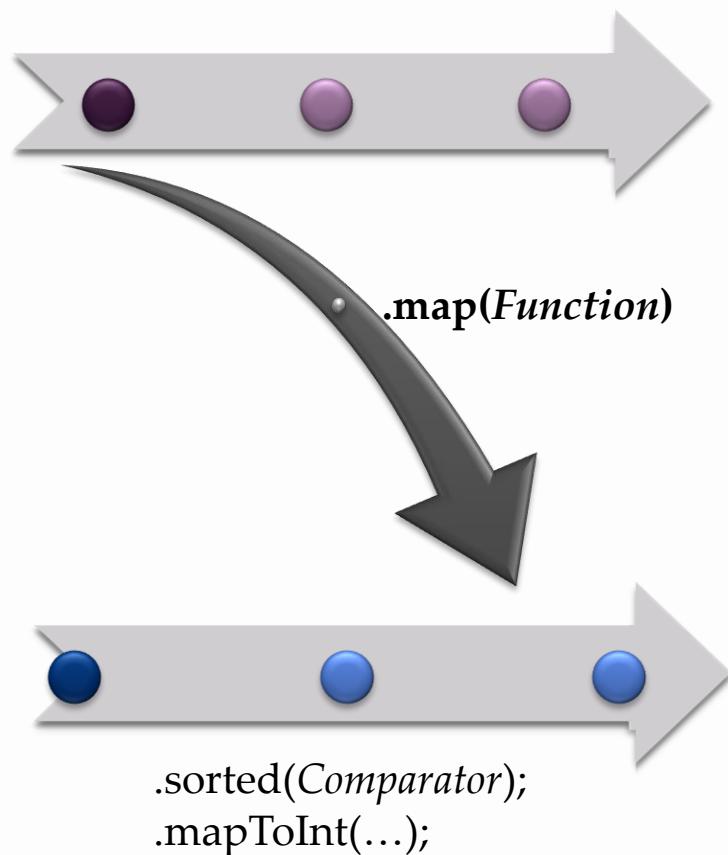
Java

Stream. Filtrado



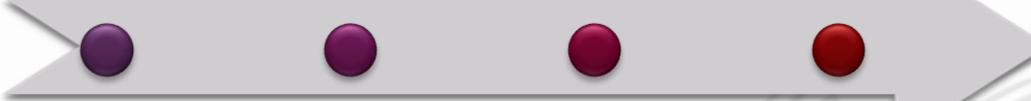
Java

Stream. Transformación

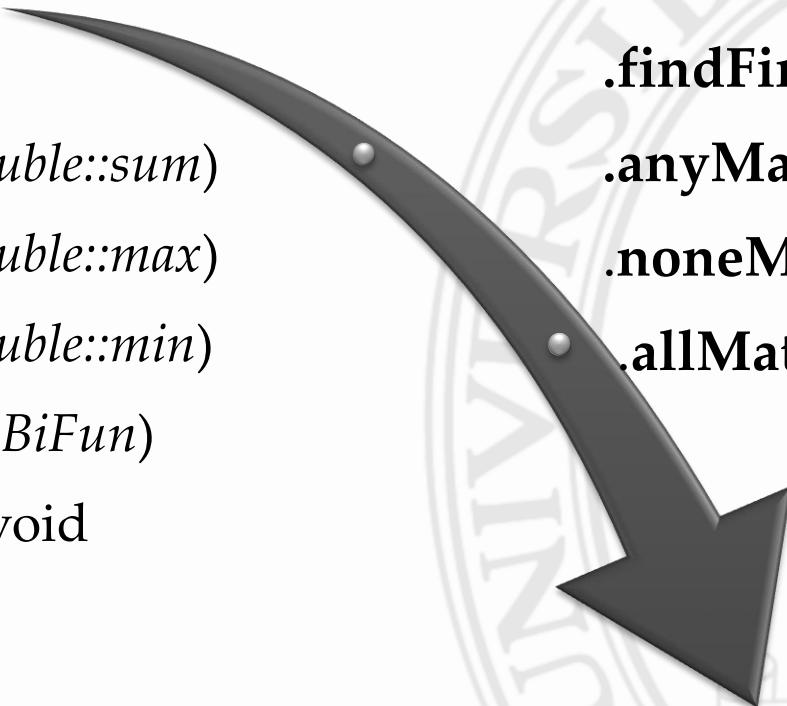


Java

Stream. Operaciones terminales



.reduce(Double::sum)
.reduce(Double::max)
.reduce(Double::min)
.reduce(ini,BiFun)
.forEach():void



.findFirst(Predicate)
.anyMatch(Predicate)
.noneMatch(Predicate)
.allMatch(Predicate)

item



<https://github.com/miw-upm/iwvg-devops>

- Package: *es.upm.miw.iwvg_devops.code*

1. *Point, DecimalCollection*

2. *Lambda, Flow*

3. *DecimalFunctionalStream, DecimalStream*

4. *User, Fraction, UsersDatabaseSeeding*

¿Qué es Apache Maven?

Es una herramienta de gestión y construcción de proyectos de software con Java.

Identificar el componente

Resolver dependencias

Test

Empaquetar

...

Se basa en un modelo de objetos del proyecto (*POM*)

Fichero *pom.xml*

Se sitúa en la raíz del proyecto.



Artefacto

Componente
software

Unidad mínima con
la que trabaja
Maven

Coordenadas

Sistema con el Maven
determina de forma única a
cada uno de sus artefactos

Group Id

- Identificación del grupo.
Normalmente se utiliza el nombre
del dominio, al revés: es.upm.miw

Artifact Id

- Identificación del artefacto: devops

Version

- 1.0.0-SNAPSHOT
- 1.3.4-RC (Release Candidate)
- 1.4.5-Release

Empaquetado

Tipo de fichero

JAR, POM

WAR, EAR, RAR...

Maven

Comandos

clean

- Elimina los ficheros generados en construcciones anteriores

validate

- Valida el proyecto si es correcto

compile

- Genera los ficheros *.class compilando los fuentes *.java

test

- Ejecuta los test unitarios (*Test) existentes

package

- Genera el empaquetado final (jar, war...)

integration-test

- Despliega el paquete y ejecuta los test de integración (*IT)

verify

- Verificar que el paquete cumpla los criterios de calidad

install

- Instala el paquete en el equipo local

deploy

- Instala el paquete en el repositorio remoto

```
C:\work-spaces\depops>mvn -v
C:\work-spaces\depops>mvn -help
C:\work-spaces\depops>mvn clean package
C:\work-spaces\depops>mvn clean -Dmaven.test.skip=true package
```

Maven Plugin

Plugin

jacoco

- Es una tarea específica, más pequeña que una fase de construcción, que contribuye a la construcción y gestión del proyecto.

sonar:sonar

- Genera un informe de cobertura
- Se conecta con *Sonarcloud* para inspeccionar el código

spring-boot:run

- Arranca una aplicación en local realizada con *Spring Boot*

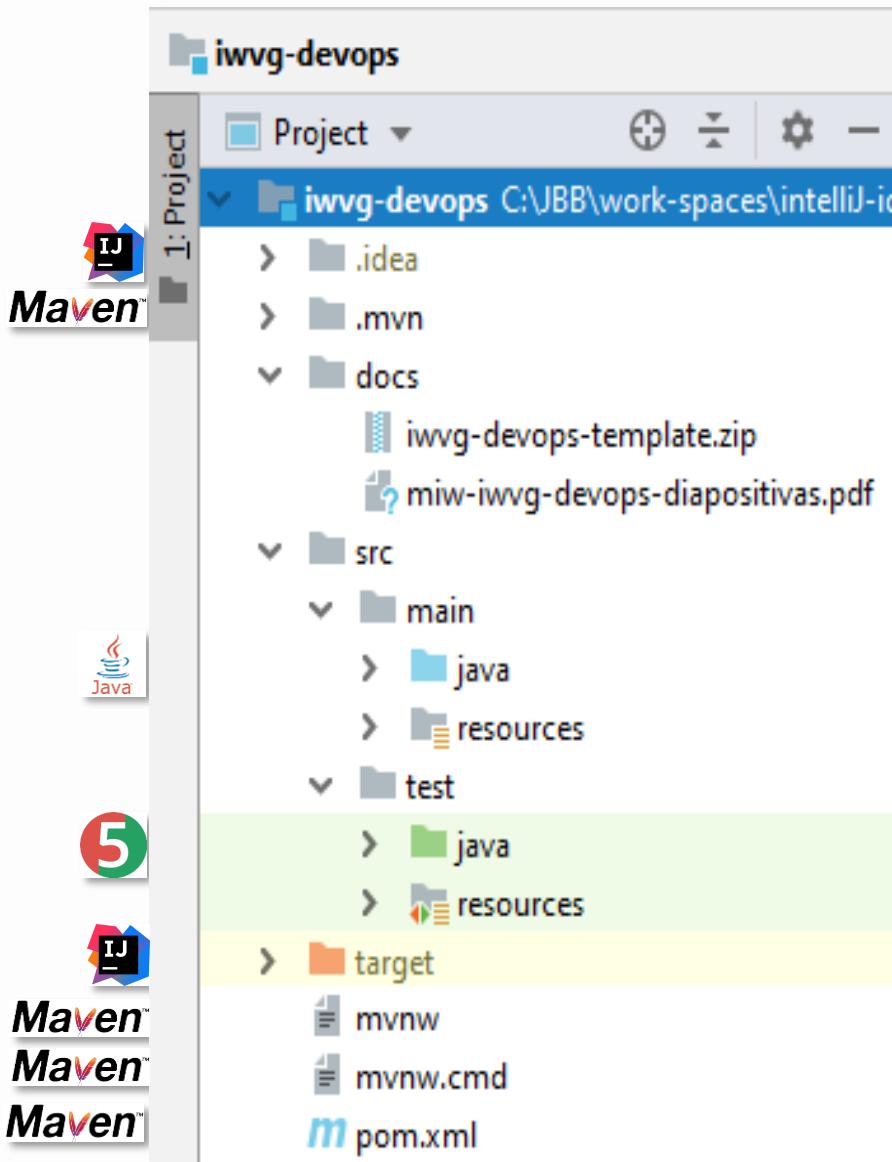
```
C:\work-spaces\betca-tpv-spring>mvn spring-boot:run
[INFO] Scanning for projects...
[INFO] -----
[INFO] < es.upm.miw:betca-tpv-spring >-----
[INFO] Building es.upm.miw.betca-tpv-spring 2.4.0-SNAPSHOT
...
...
```

Maven POM

```
m iwg-devops x
1 <?xml version="1.0" encoding="UTF-8" ?>
2 <project
3     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4     xmlns="http://maven.apache.org/POM/4.0.0"
5     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
6     <modelVersion>4.0.0</modelVersion>
7
8     <parent...>
14
15     <artifactId>iwg-devops</artifactId>
16     <groupId>es.upm.miw</groupId>
17     <version>1.4.0-SNAPSHOT</version>
18     <packaging>jar</packaging>
19
20     <name>${project.groupId}.${project.artifactId}</name>
21     <description>DevOps</description>
22     <url>http://github.com/miw-upm/${project.artifactId}</url>
23
24     <licenses...>
30
31     <developers...>
43
44     <properties...>
66
67     <dependencies...>
110     <build>
111         <plugins...>
178     </build>
179 </project>
```

Maven

Estructura del proyecto



Instalación externa Maven

Bajarse y descomprimir:

- *-bin.zip

Variable de entorno

- **M2_HOME** (al raíz).
- **PATH** (al bin).

Se necesita tener instalado el **OpenJDK** o **JDK**. Variables de entorno:

- **JAVA_HOME**: Al raíz.
- **PATH**: Al bin

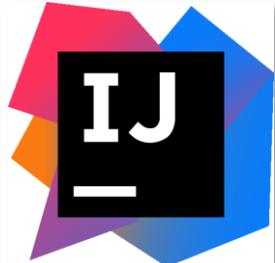
Maven IDE

IntelliJ IDEA (JetBrains)

- IDE para Java
- Tiene una versión gratuita: *Community Edition*.

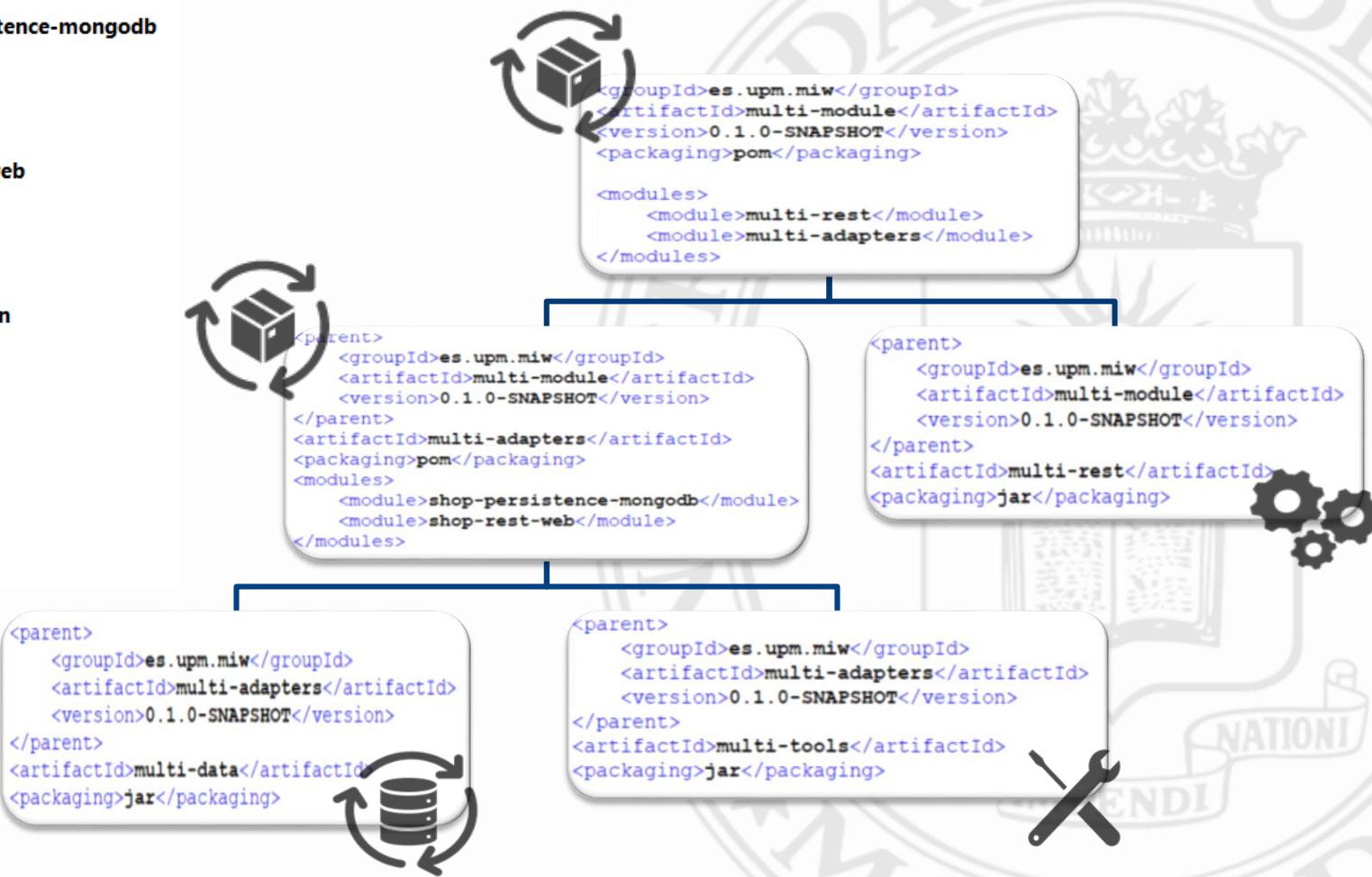
Crear un proyecto nuevo

- Partir de un *Template*
- <https://github.com/miw-upm/iwvg-devops/docs>
 - Descomprimirla y cambiar coordenadas
- *Importar desde IntelliJ*, indicando que es de tipo *Maven*.



Maven Multi-módulo

```
apaw-shop-hexagonal
  .idea
  docs
  shop-adapters
    shop-persistence-mongodb
      .idea
      src
      pom.xml
    shop-rest-web
      src
      pom.xml
      pom.xml
  shop-application
    src
    pom.xml
  shop-domain
    .gitignore
    .travis.yml
    LICENSE.md
    pom.xml
```



Maven

Características

Repositorio

- Estructura de directorios y archivos que usa Maven para almacenar, organizar y recuperar artefactos.
- Existen repositorios *locales, privados y remotos*
 - Repositorio central de Maven: <https://mvnrepository.com/>
 - Repositorio local: `%User%/.m2/repository`.

Perfiles

- Permite cambiar la configuración de la aplicación dependiendo del entorno en el que se despliega.
- Se definido en el archivo `settings.xml`. y se utiliza en el `pom.xml`

Arquetipos

- Plantilla para crear proyectos.

Maven



1. Instalar *OpenJDK 11* de Java. Definir variables de entorno.

2. *Maven*. Se manejará maven desde la consola.

- Instalar ***Maven***. Definir variables de entorno o Meter *Maven* en el proyecto. Utilizar el *Maven* embebido.

3. Instalar IntelliJ IDEA.

4. Crear la carpeta de los *workspaces*.

5. Crear un proyecto Java con *maven* en el *workspace*

- Se ofrece una plantilla a modo de ejemplo: <https://github.com/miw-upm/iwvg-devops/docs>.
- Recordar cambiar el **nombre de la carpeta** y del **artefacto** en el fichero *pom.xml*.

6. Importar el proyecto desde *IntelliJ IDEA*.

- Cerrar proyecto si estuviese abierto.*
- Import Project*, y seleccionar la carpeta del proyecto.
- Marcar *Create Project from external model*, elegir *Maven* .
- Next... Finish.*

7. Probar comandos.

- Maven* embebido: ***maven: mvnw -v, mvnw package...***
- Maven* externo: ***maven: mvn -v, mvn package...***

Test Tipos

Pruebas Unitarias (**Test)

- Se prueba un módulo de código o clase independientemente del resto. Si existen dependencias entre componentes se rompen con los *mocks*.

Pruebas de Integración (**IT)

- Se prueban los diferentes componentes que dependen de otros componentes.

Pruebas Funcionales (**FT)

- Se prueba el sistema como un todo.

Pruebas de Aceptación

- Los clientes prueban la versión entregada.

Test

Característica de calidad

Automática	Integración Continua	Cobertura	Independiente	Sencillez
<ul style="list-style-type: none">• Clases que prueban clases.	<ul style="list-style-type: none">• Cuando parte del código ha sido modificado, se vuelven a lanzar todas las pruebas.	<ul style="list-style-type: none">• % de líneas de código ejecutadas en las pruebas.• >80%.	<ul style="list-style-type: none">• El orden de las pruebas es independiente.• Las pruebas no alteran el sistema.• Se prueban los módulos por separado.	<ul style="list-style-type: none">• Prueba una cosa a la vez.

Test

JUnit 5

JUnit

Es un framework que nos ayuda a la realización de pruebas unitarias

Fue creado por *Erich Gamma* y *Kent Beck*.

Sub-proyectos

Platform

Es el responsable de lanzar los test sobre la JVM

Jupiter

Es el nuevo modelo de programación

Vintage

Es para compatibilizar JUnit3 y 4 sobre JUnit5

Test JUnit 5

@BeforeAll

- Se ejecuta una sola vez antes de la batería de pruebas definida en la clase y el método debe ser *static*

@BeforeEach

- Se ejecuta antes de cada uno de los marcados con @. Suele ser una inicialización por todas las pruebas de la clase

@Test

- Marca un método como prueba

@AfterEach

- Se ejecuta después de cada uno de los @Test. Suele ser una liberación de recursos

@AfterAll:

- Se ejecuta al final del proceso completo y el método debe ser static

@Test
assert**

assertEquals, assertNotEquals,
assertArrayEquals, assertTrue,
assertFalse, assertEquals,
assertNotSame, assertNull,
assertNotNull, fail(),
assertThrows,
assertDoesNotThrow...



<https://github.com/miw-upm/iwvg-devops>

- Package: `es.upm.miw.iwvg-devops.code`

1. Ejemplos de test

- Point: `PointTest`, DecimalCollection: `DecimalCollectionTest`
- `DecimalFunctionalStreamTest`, `DecimalStreamTest`
- `FlowTest`, `searchesTest`

2. Lanzamiento de test con IntelliJ

3. Ejercicios: Crear los test para el resto de clases del paquete

- User, Fraction
- ...

Git

Sistema de Control de Versiones

Tipos

- **Distribuidos.** Aumenta la flexibilidad pero complica la sincronización y gestión. Ejemplos: *Git, Mercurial...* En la actualidad se están imponiendo estos sistemas.
- **Centralizados.** Dependiente de un responsable. Facilita la gestión pero reduce la potencia y flexibilidad.

Git

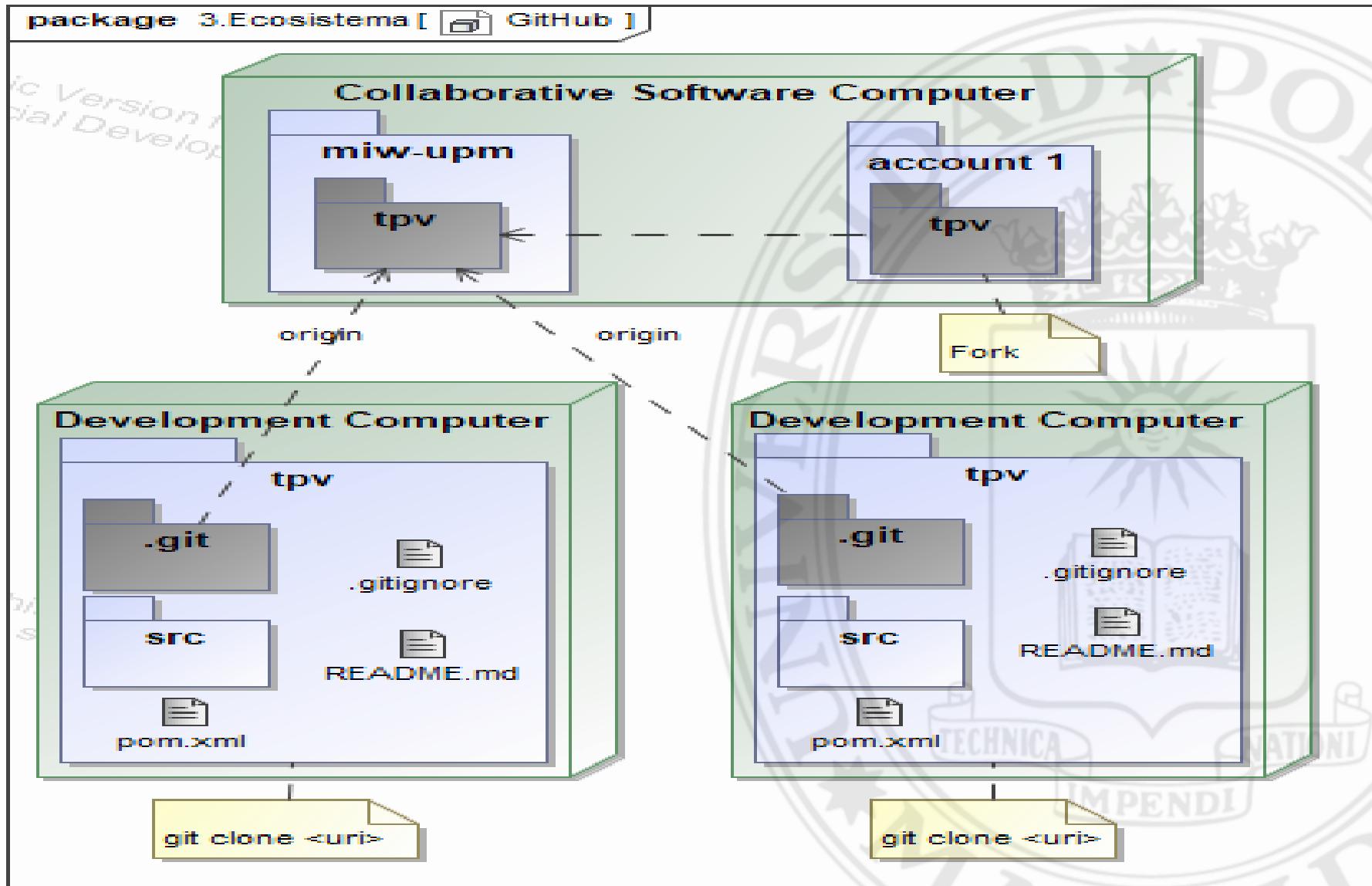
- Nace en 2005.
- Toma como experiencia el proyecto Bitkeeper. (propietario).
- En 2008 nace GitHub, Git Cloud.
- En el 2018 Microsoft adquiere GitHub por 7.500 millones de dólares.
- Cliente: <https://git-scm.com>

Características

- Control de versiones distribuido.
- Muy fiable, imposible perder el proyecto.
- Trabaja sin necesidad de conexión al remoto, muy rápido. Se podrá sincronizar con el remoto, pero con asistencia...
- **Snapshot:** instantánea (commits)

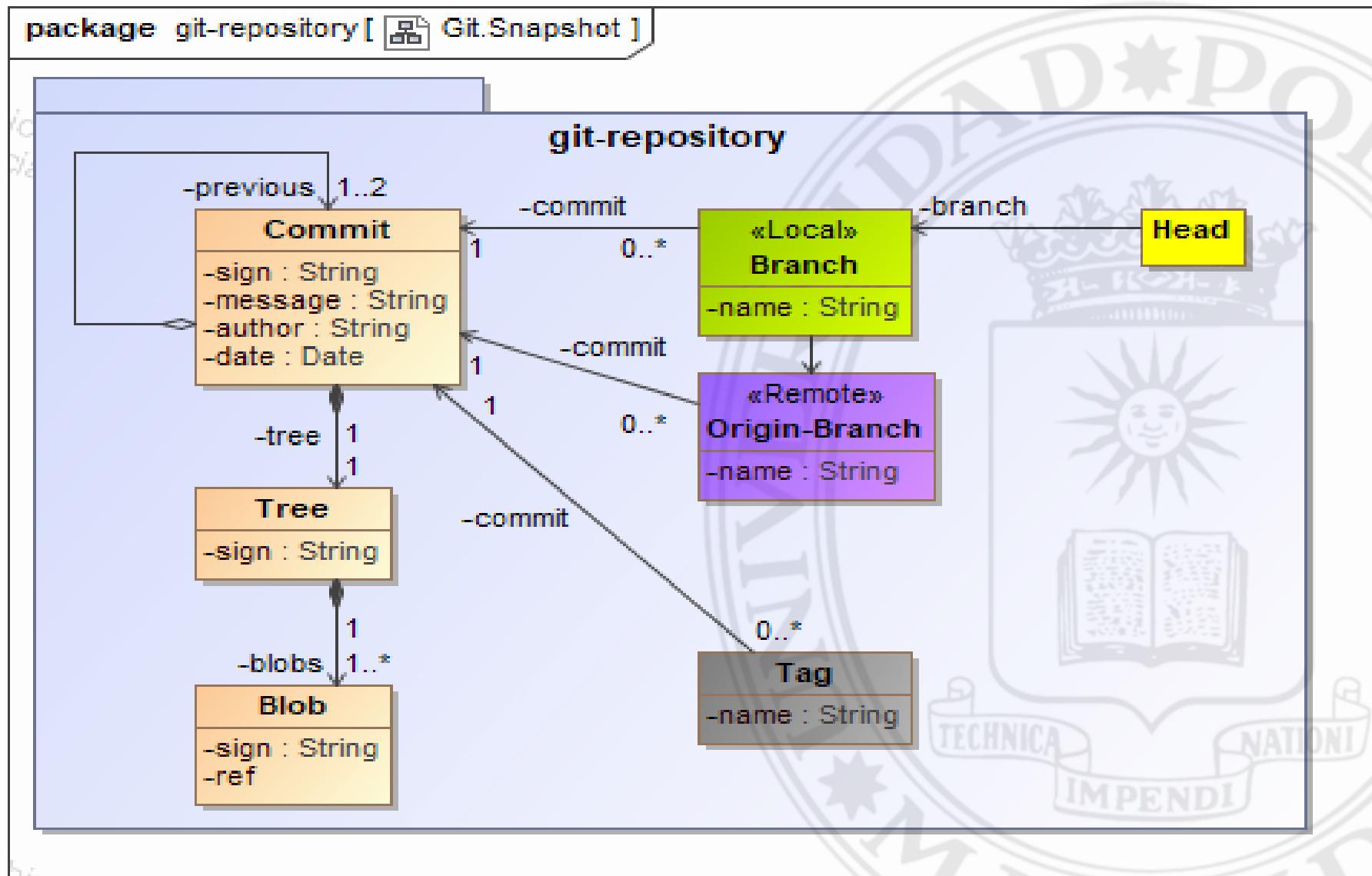
Git

Sistema de control de versiones



Git

Instantánea (snapshot)



Git IntelliJ

The screenshot shows the IntelliJ IDEA interface with the Version Control tab selected. A red arrow points to the 'Log' tab in the top navigation bar. The main area displays a git log with the following entries:

Commit Message	Branches	Date	Author
reformat all code	origin & develop	19/05/2019 17:20	jbernal
update Swagger page description #1 on develop	origin & develop	19/05/2019 16:50	jbernal
version 1.0.2 & Heroku Procfile	origin & master	19/05/2019 16:31	jbernal
Merge bug #1 into release-1.0	origin & master	19/05/2019 16:22	jbernal
update swagger page description #1	origin & release-1.0	19/05/2019 16:19	jbernal
add mongodb embedded test scope	master, release-1.0	19/05/2019 13:19	jbernal
version 1.0.1	origin/master, origin/release-1.0	19/05/2019 13:00	jbernal
update sprig-boot version to 2.1.5	release-1.0.2	19/05/2019 11:34	jbernal
version 1.1.0-SNAPSHOT		19/05/2019 11:30	jbernal
prepare release: version 1.0.0 & prod profile		19/05/2019 11:29	jbernal
heroku add Procfile		19/05/2019 8:20	jbernal
heroku add deploy		19/05/2019 8:17	jbernal
better-code-hub depth to 7		19/05/2019 7:33	jbernal
ecosystem add sonarcloud & better-code-hub		19/05/2019 7:23	jbernal
ecosystem add Travis-CI		19/05/2019 7:07	jbernal
add docs & readme		19/05/2019 6:55	jbernal
Initial TPV		19/05/2019 6:40	jbernal

A red arrow points to the commit entry "ecosystem add Travis-CI". To the right of the log, a file browser shows a folder named "etsisi-tpv-spring" containing ".travis.yml" and "README.md". A red arrow points to the ".travis.yml" file.

At the bottom, a red arrow points to the "Version Control" tab in the footer navigation bar.

Footer tabs: Version Control, Terminal, Java Enterprise, Spring, Messages, Find, Run, TODO.

Git

Comandos

help

- git help --all

status

- git status

config

- git config --global user.name "..."
- git config --global user.mail "..."
- git config credential.helper store
- git config --list

clone

- git clone <uri>

Git

Comandos

checkout

- `git checkout -b <branch>`
- `git checkout <branch>`

commit

- `git commit -m "message #666"`
- `git commit --amend --no-edit`
- `git commit --amend -m "new message"`

merge

- `git merge -m "merge develop into #666" develop`
- `git merge --no-ff -m "merge #666 into develop. Details" issue#666`
- `git merge origin/develop`

reset

- `git reset --hard HEAD`
- `git reset --hard <commit>`
- Borrado de rama y creación en el commit correcto

init

- `git ini`
- `git add --all`
- `git commit -m "mi mensaje de commit inicial"`

Clonar el repositorio

- <https://github.com/miw-upm/iwvg-devops>

Importar desde IntelliJ IDEA

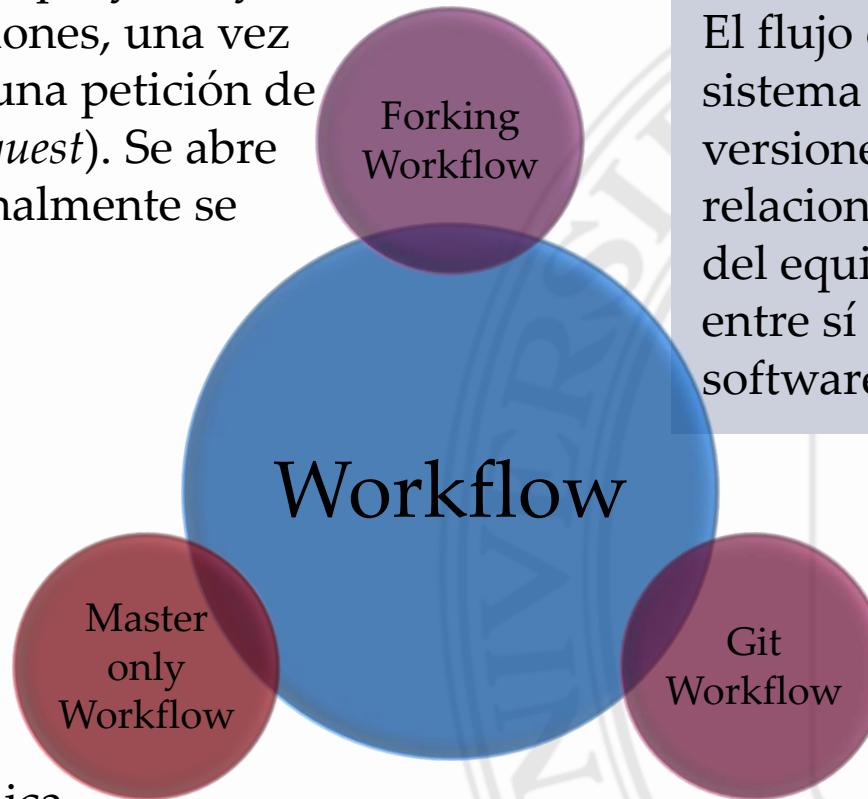
Demo in Action

- En una carpeta de trabajo propia, clonar repositorio
<https://github.com/miw-upm/iwvg-git-workflow>
- Branches exercises: Note 0
- Proyecto: <https://github.com/miw-upm/iwvg-git-workflow/projects>

Git

Flujo de Trabajo

El usuario bifurca el proyecto y realiza las ampliaciones, una vez finalizado, realiza una petición de agregación (*pull request*). Se abre una discusión, y finalmente se fusión.



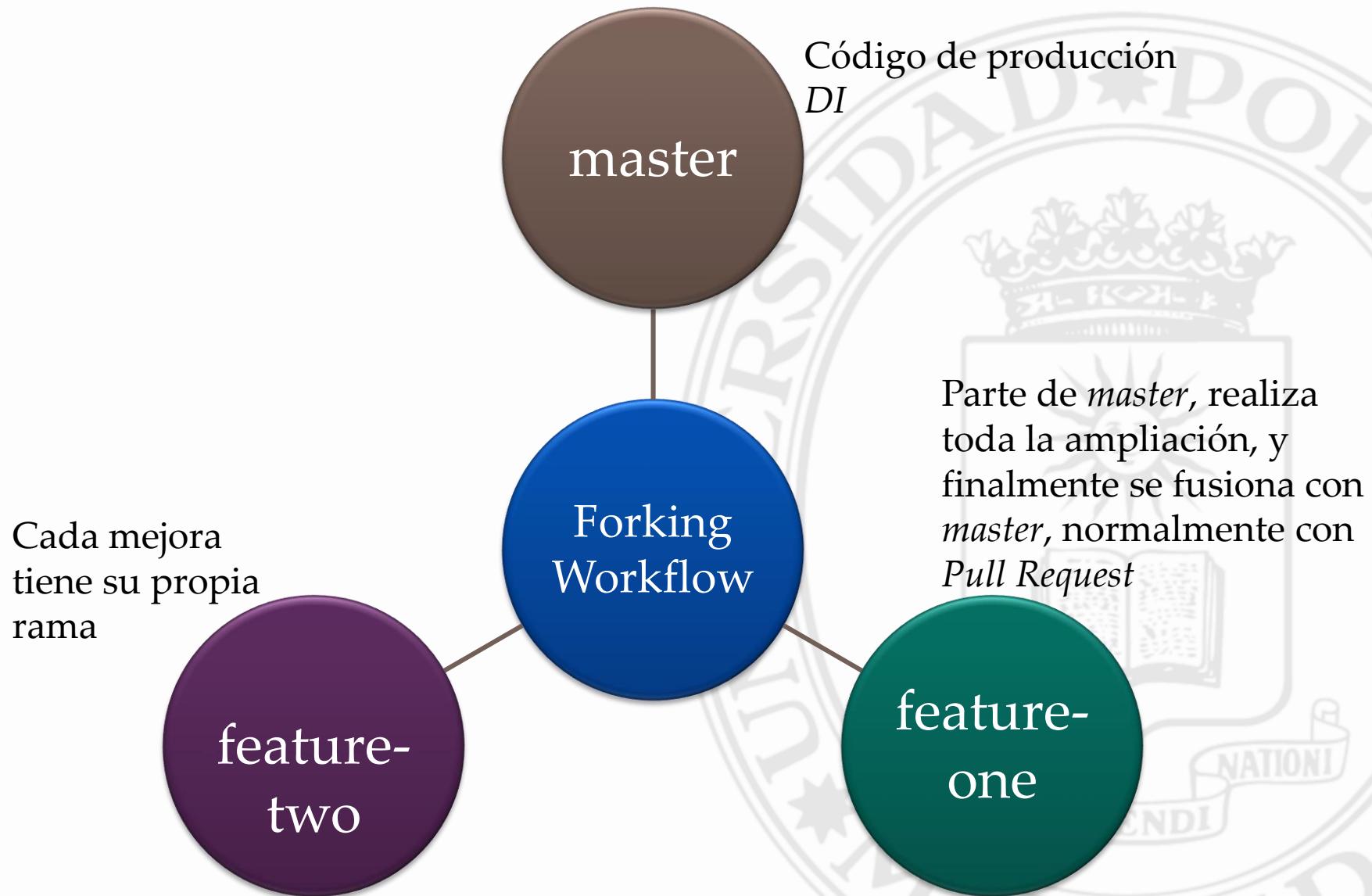
Todos comparten la única rama... Fusión problemática!!!

El flujo de trabajo de un sistema de control de versiones indica cómo se relacionan los miembros del equipo para colaborar entre sí en el desarrollo del software colaborativo

Existen varias ramas con distintas funcionalidades dentro del equipo: *master*, *develop*, *feature*, *release*, *bug*

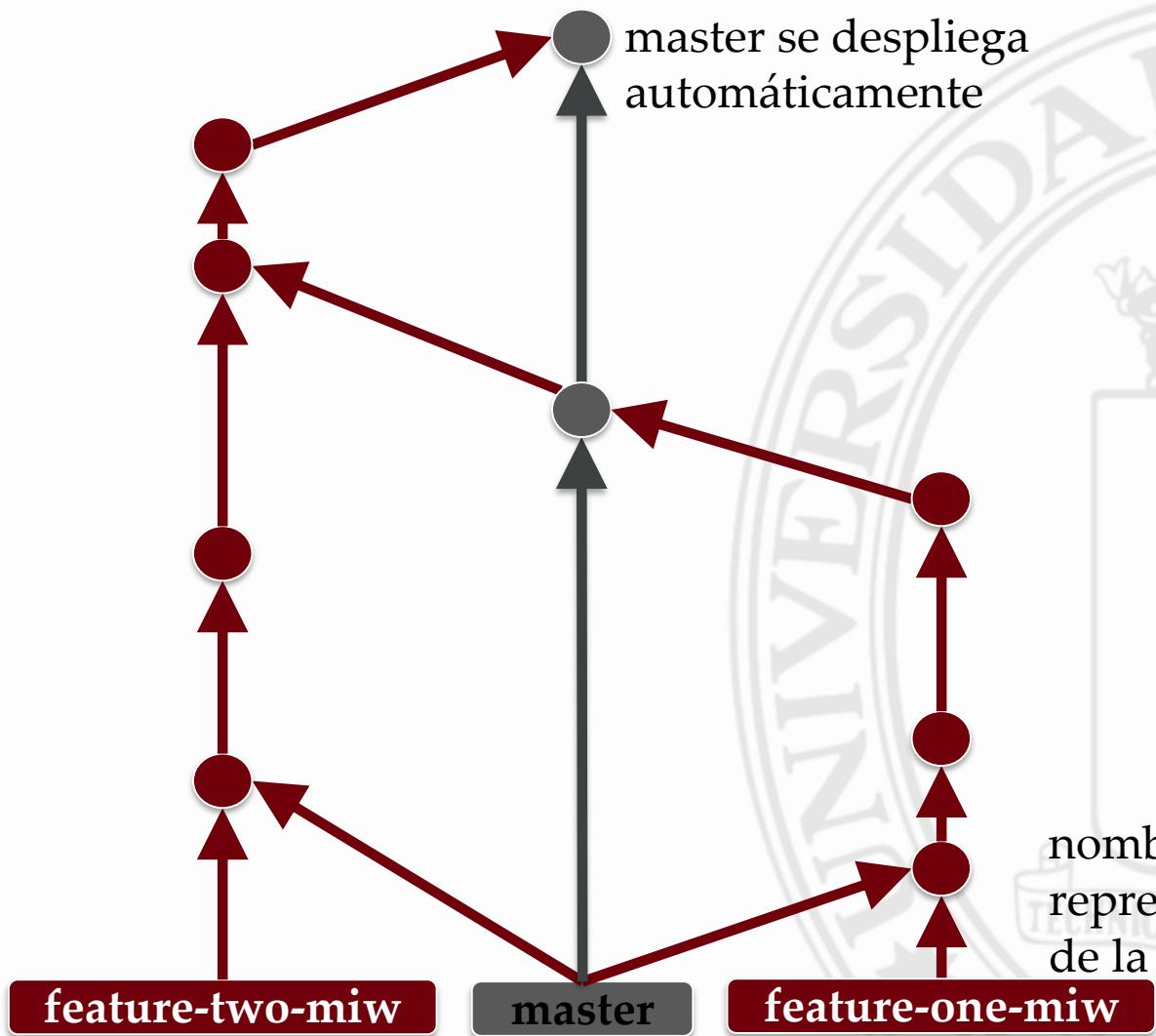
Git

Flujo de Trabajo Bifurcación



Git

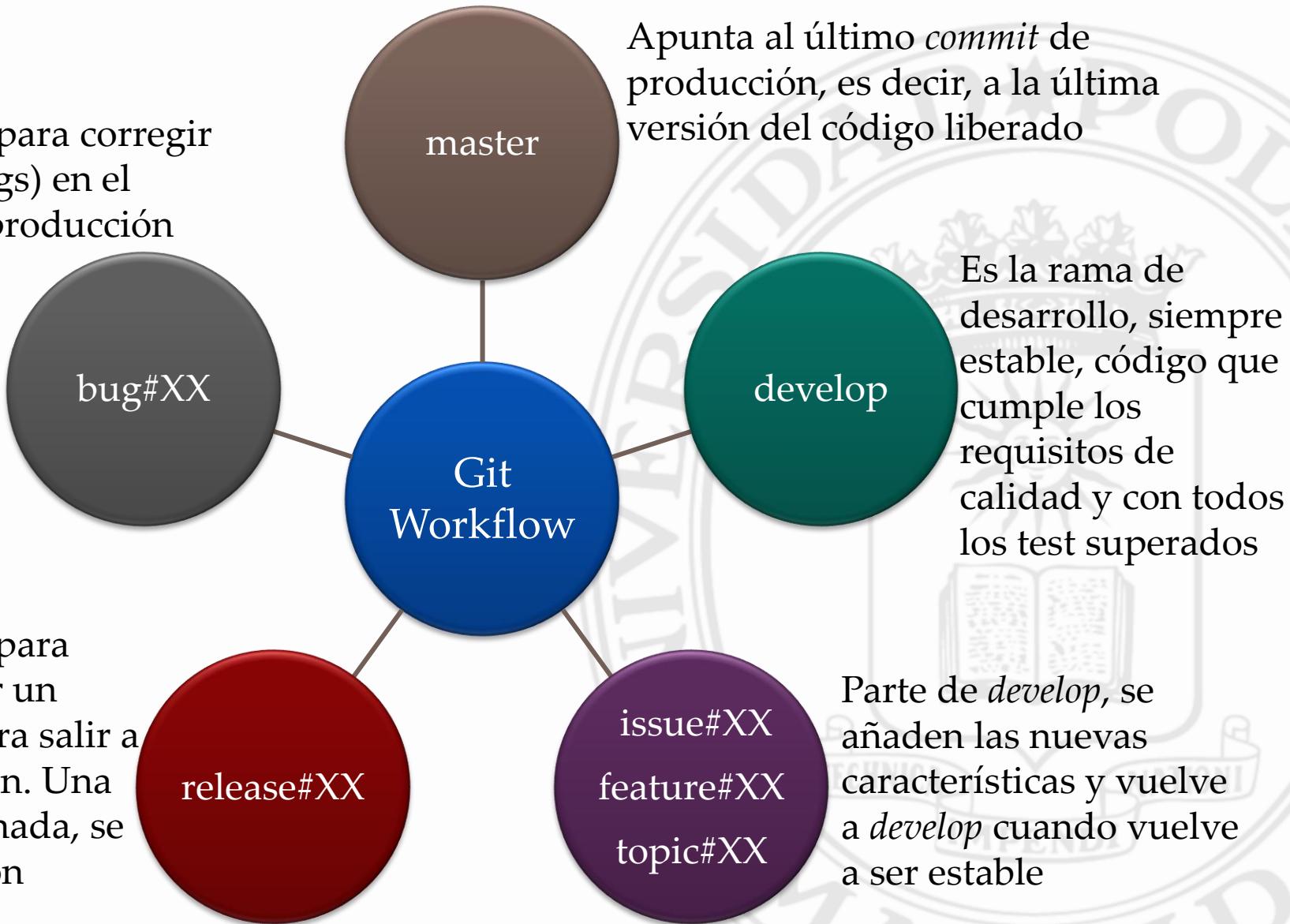
Flujo de Trabajo Bifurcación



Git

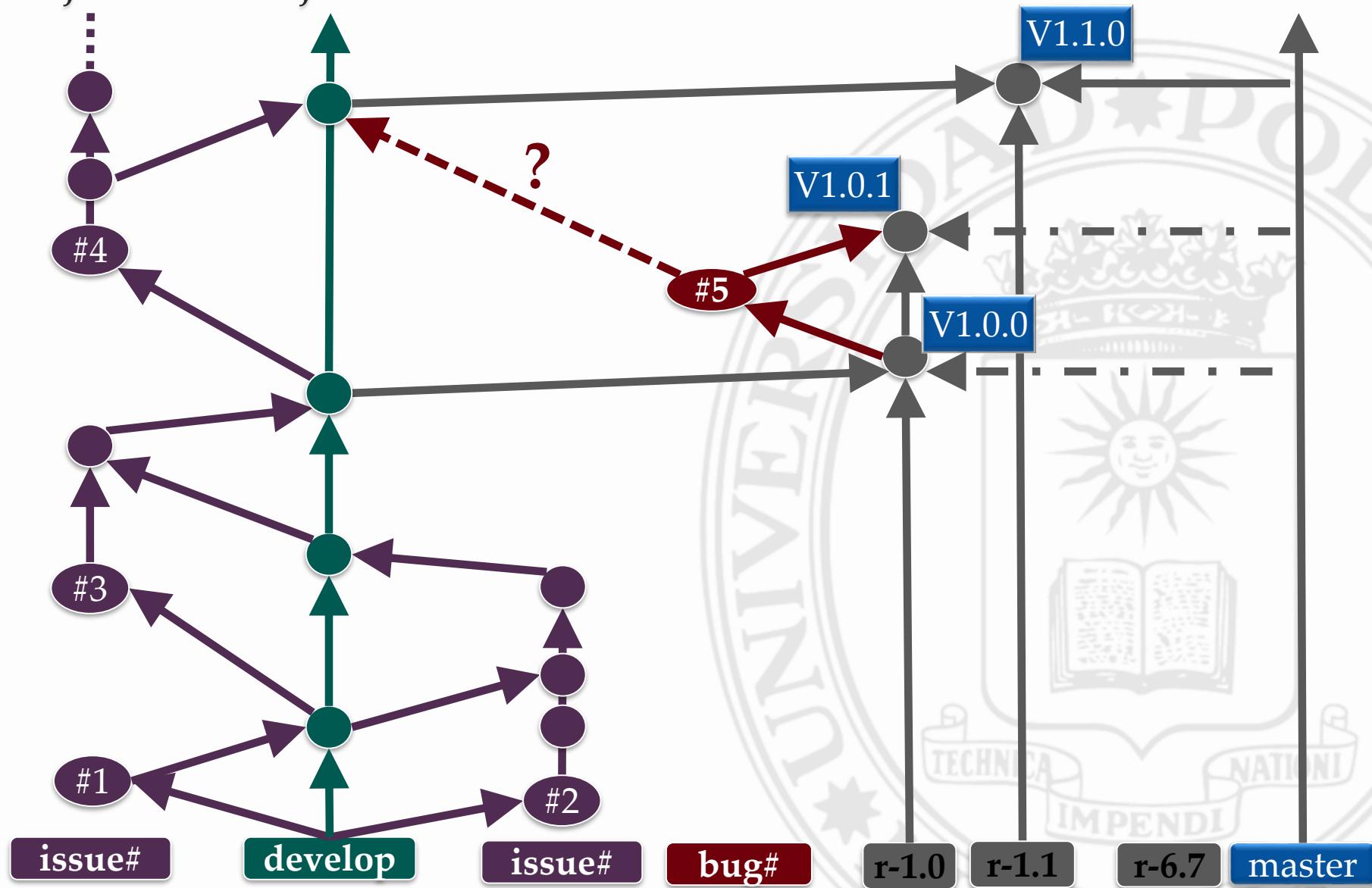
Flujo de Trabajo Ramificado

Se utilizan para corregir errores (bugs) en el código en producción



Git

Flujo de Trabajo Ramificado



Git

Workflow: local

add C2:m1 #16

add C1:m1 #16

refactoring package names



inicio
issue#

- git checkout -b issue#16
- git add -all
- git commit -m "add C1:m1 #16"
- git commit -m "... #16"

Merge issue #16 into develop

add C2:m1 #16

add C1:m1 #16

refactoring package names



aportación
parcial

- git checkout develop
- git merge --no-ff -m "... issue#16
- git checkout issue#16

Merge issue #16 into develop

add C2:m2 #16

add C1:m2 #16

Merge issue #16 into develop

add C2:m1 #16

add C1:m1 #16

refactoring package names

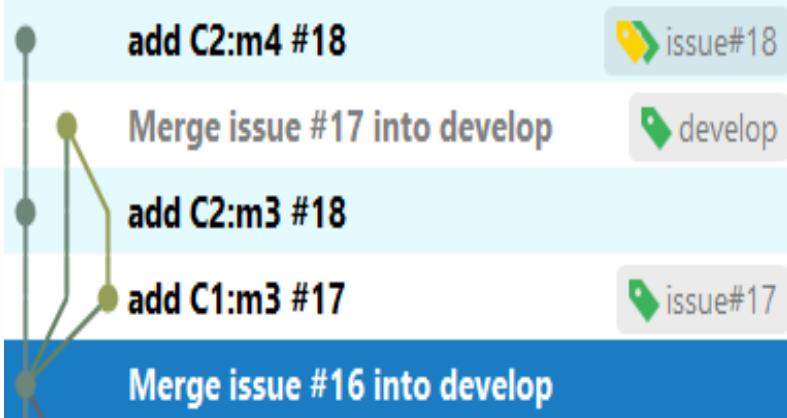


cierre
issue#

- git commit -m "..."
- git commit -m "..."
- git checkout develop
- git merge --no-ff -m "... #16" issue#16

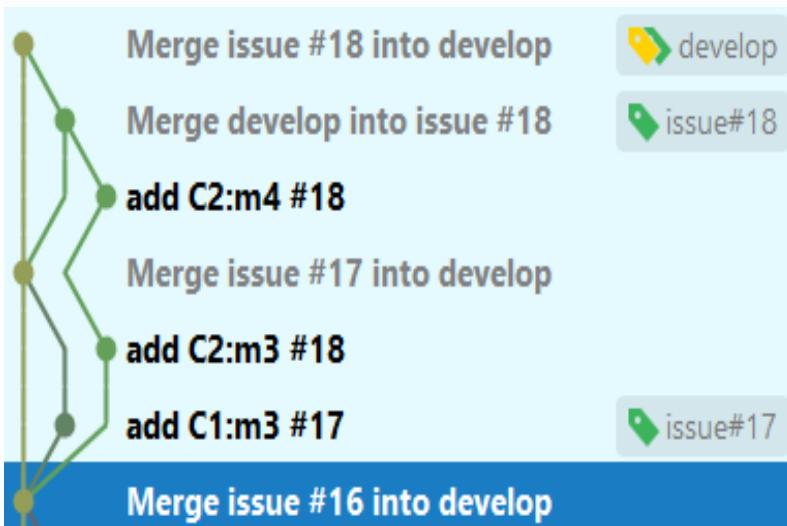
Git

Workflow: local



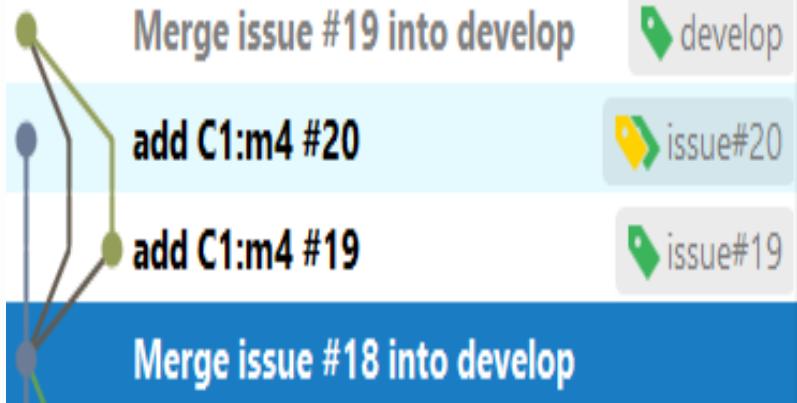
cierre
issue#18
SIN
conflicto

- git merge -m "..." develop
- git checkout develop
- git merge --no-ff -m "..." issue#18



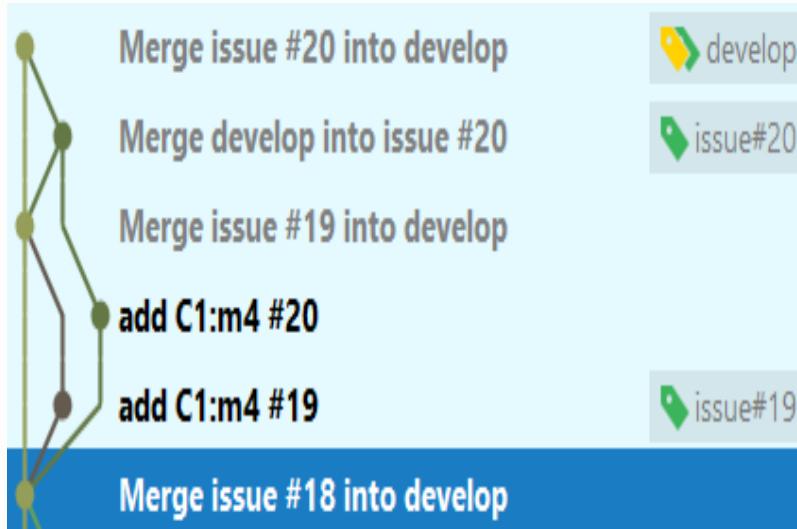
Git

Workflow: local



cierre
issue#18
CON
conflicto

- git merge -m "..." develop
- *Editar C1 y dejar su contenido correcto*
- git add --all
- git commit
- git checkout develop
- git merge --no-ff -m "..." issue#18



Git. Fusiones



✍ Demo in Action

- *Sin conflictos*
- *Con conflictos*

✍ Workflow exercises: Note 1

- Proyecto: <https://github.com/miw-upm/iwvg-git-workflow/projects>

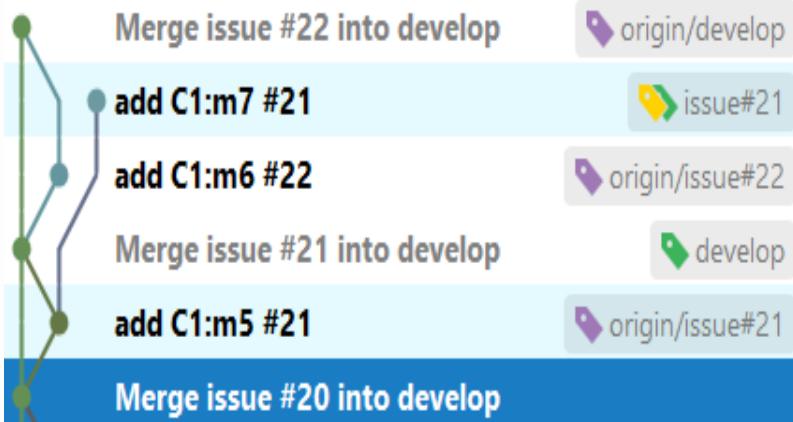
Git

Workflow: remoto



subir
develop

- Solo FF (FastForward)
- git push origin develop

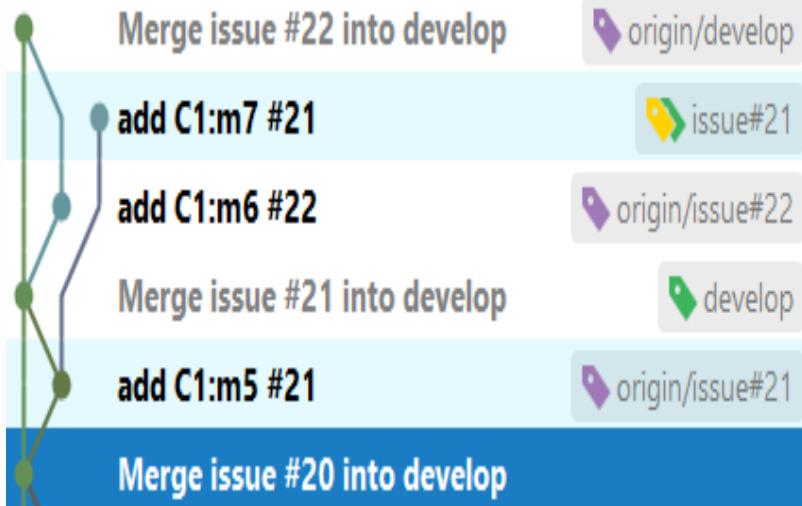


cierre
issue#21

- git fetch origin
- ???

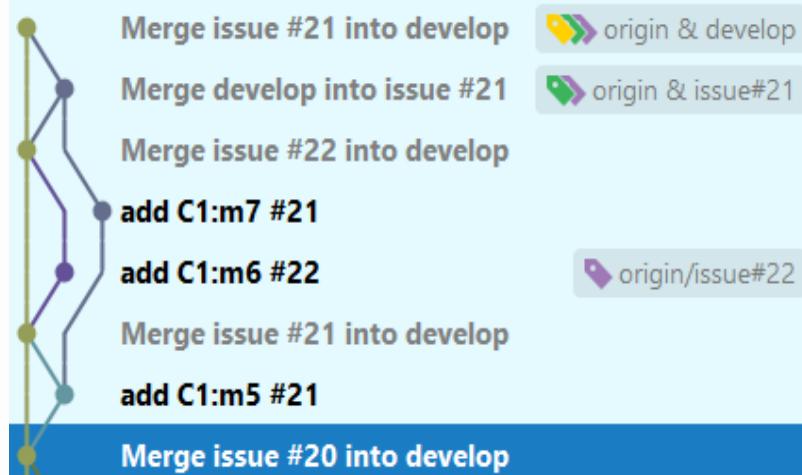
Git

Workflow: remoto



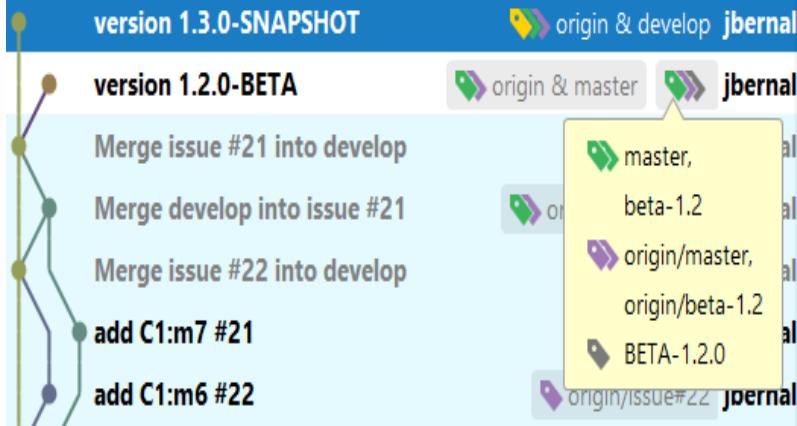
cierre
issue#21

- git checkout develop
- git merge origin/develop
- git check issue#21
- git merge -m "... develop
- git checkout develop
- git merge --no-ff -m "... issue#21
- git push origin develop



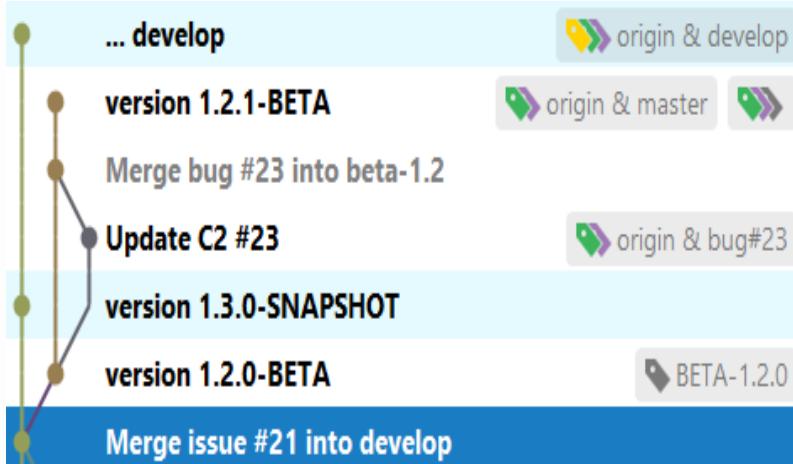
Git

Workflow: release



subir
develop

- git tag -a BETA-1.2.0 -m "..."
- git push origin BETA-1.2.0
- git branch -d master
- git branch master
- git push origin master --force
- git checkout develop
- git commit -m "."
- git push origin develop



cierre
issue#21

- git checkout beta-1.2
- git checkout -b bug#23
- git commit -m "..."
- git checkout beta-1.2
- git merge --no-ff -m "..." bug#23
- git push origin beta-1.2
- Liberar nueva release

Git

Workflow: test



Errores?

• Solución?



Error?

• Solución?

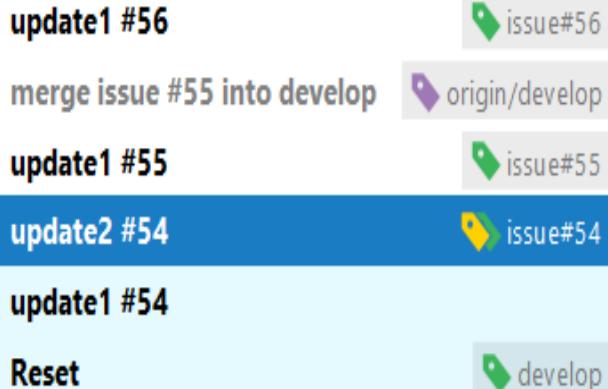


Error?

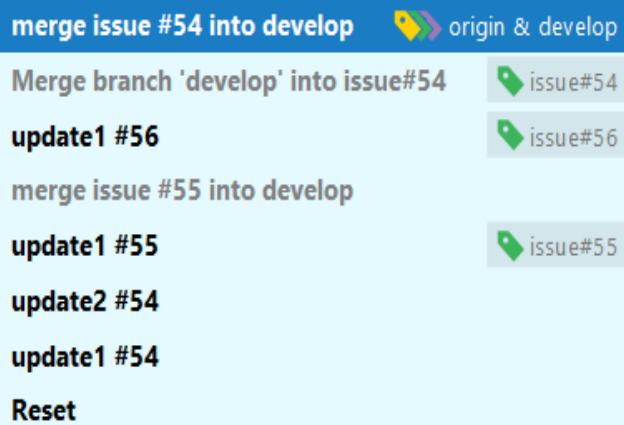
• Solución?

Git

Workflow: test



cierre
issue#54



Error?

- Quién #54, #55 o #56?
- Error cometido?
- Solución?

Git

Comandos

Repositorios remotos

- Consultar los repositorios remotos: **git remote -v**
- Añadir repositorio remoto: **git remote add origin <url>**
- Eliminar un repositorio remoto: **git remote rm origin**
- Subir una rama al remoto: **git push origin <branch>**
- Subir todas las ramas: **git push origin --all**
- Bajarse todas las ramas remotas: **git fetch origin**
- Bajarse una rama: **git fetch origin <branch>**

Etiquetas

- Consulta de etiquetas: **git tag**
- Crear etiqueta: **git tag -a <etiqueta> -m "mensaje"**
- Borrado de etiqueta: **git tag -d <etiqueta>**
- Subir etiqueta al remoto: **git push origin <etiqueta>**

Recuperación de situaciones anómalas, conlleva riesgo de pérdida de código

- Incorpora los nuevos cambios al último commit: **git commit --amend --no-edit**
- Cambiando el mensaje del último commit: **git commit --amend -m "new message"**
- Volver a un commit: **git reset --hard HEAD**, **git reset --hard <commit>**
- Subir una rama forzando el cambio: **git push origin <branch> --force**

Git. Workflow

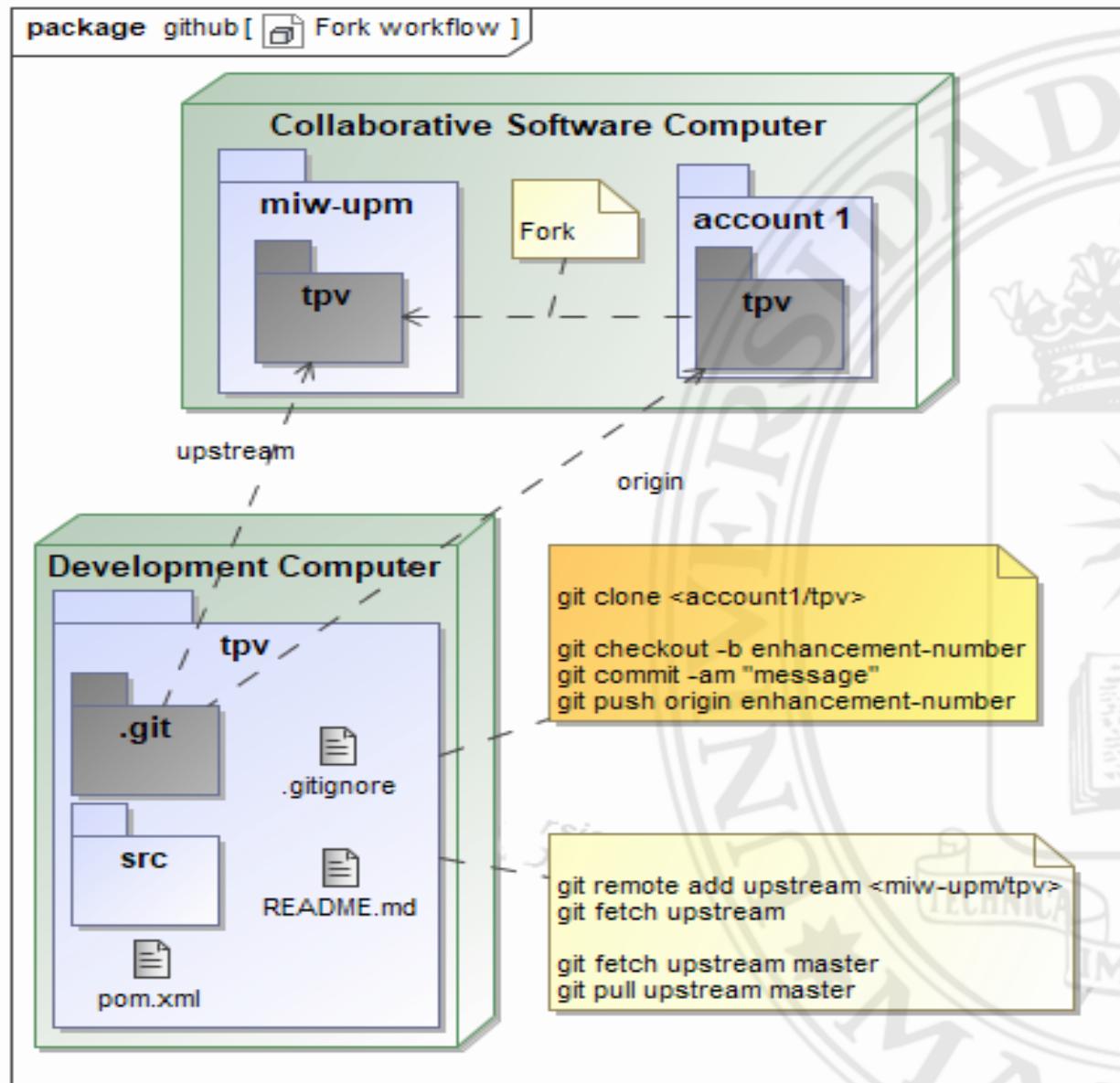


✍ Workflow exercises: Note 2

- Proyecto: *<https://github.com/miw-upm/iwvg-git-workflow/projects>*

GitHub

Flujo de Trabajo Bifurcación



GitHub

Flujo de Trabajo Bifurcación

Forking Workflow

- **git fetch upstream / git pull upstream (con frecuencia)**
- **Nombres de ramas descriptivos**

jesusBernalBermudez / **iwvg-fork-workflow**
forked from miw-upm/iwvg-fork-workflow

Unwatch ▾ 1 Star 0 Fork 1

Code

Pull requests 0

Projects 0

Wiki

Insights

Settings

Universidad Politécnica de Madrid. Máster en Ingeniería Web. IWVG

Edit

Manage topics

2 commits

2 branches

0 releases

1 contributor

Your recently pushed branches:

enhancement-number-one (5 minutes ago)

Compare & pull request

GitHub

Flujo de Trabajo Bifurcación

Enhancement number one #1

[Open](#) jesusBernalBermu... wants to merge 2 commits into `miw-upm:master` from `jesusBernalBermudez:enhancement-number-one`

Conversation 1 Commits 2 Checks 0 Files changed 2

JesusBernalBermu... commented 4 minutes ago

First-time contributor + ...

Enhancement number one Description

miw-upm added some commits 19 minutes ago

- miw-upm add C1
- miw-upm add C2

miw-upm requested changes just now

View changes

miw-upm left a comment

Owner + ...

Add more methods

Add more commits by pushing to the `enhancement-number-one` branch on `jesusBernalBermudez/iwvg-fork-workflow`.

Changes requested
1 review requesting changes [Learn more.](#)

Hide all reviewers

miw-upm requested changes Approve changes Dismiss review Re-request review

Continuous integration has not been set up
Several apps are available to automatically catch bugs and enforce style.

This branch has no conflicts with the base branch
Merging can be performed automatically.

Merge pull request You can also open this in GitHub Desktop or view command line instructions.

- Comment
Submit general feedback without explicit approval.
- Approve
Submit feedback and approve merging these changes.
- Request changes
Submit feedback that must be addressed before merging.

Reviewers

miw miw-upm

Assignees

No one—assign yourself

Labels

None yet

Projects

None yet

Milestone

No milestone

Notifications

Unsubscribe

You're receiving notifications because you commented.

2 participants



Lock conversation

**Login del oficial
Se puede configurar
y administrar la
petición**

Create a merge commit

All commits from this branch will be added to the base branch via a merge commit.

Squash and merge

The 3 commits from this branch will be combined into one commit in the base branch.

Rebase and merge

The 3 commits from this branch will be rebased and added to the base branch.

GitHub

Flujo de Trabajo Bifurcación

Enhancement number one #1

Merged miw-upm merged 3 commits into miw-upm:master from jesusBernalBermudez:enhancement-number-one 10 minutes ago

Conversation 1 Commits 3 Checks 0 Files changed 2 +24 -0

jesusBernalBermu... commented 26 minutes ago Contributor + ...

Enhancement number one Description

miw-upm added some commits 41 minutes ago

MIW add C1 0c4b1d6
MIW add C2 0ac3bb0

miw-upm requested changes 22 minutes ago View changes

miw-upm left a comment Owner + ...

Add more methods

MIW update C1&C2 b764964

miw-upm merged commit 08096c2 into miw-upm:master 10 minutes ago Revert

Pull request successfully merged and closed Delete branch

You're all set — the jesusBernalBermudez:enhancement-number-o... branch can be safely deleted.

If you wish, you can also delete your fork of miw-upm/iwvg-fork-workflow.

Reviewers MIW miw-upm

Assignees No one assigned

Labels None yet

Projects None yet

Milestone No milestone

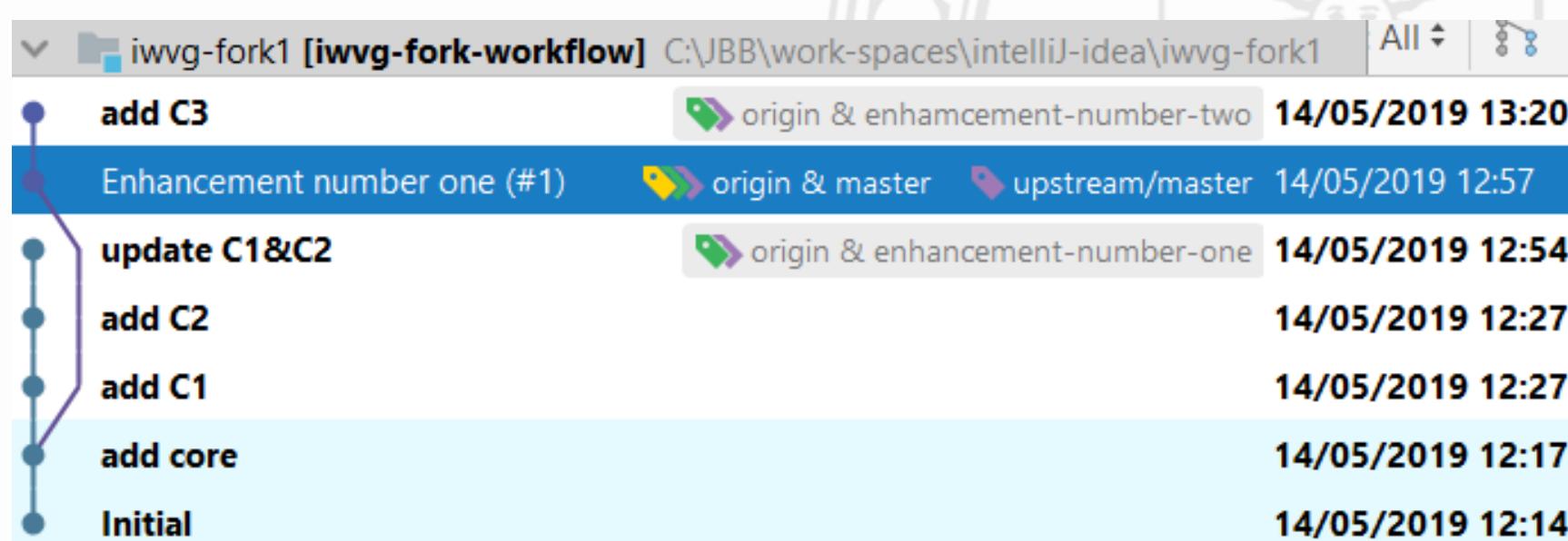
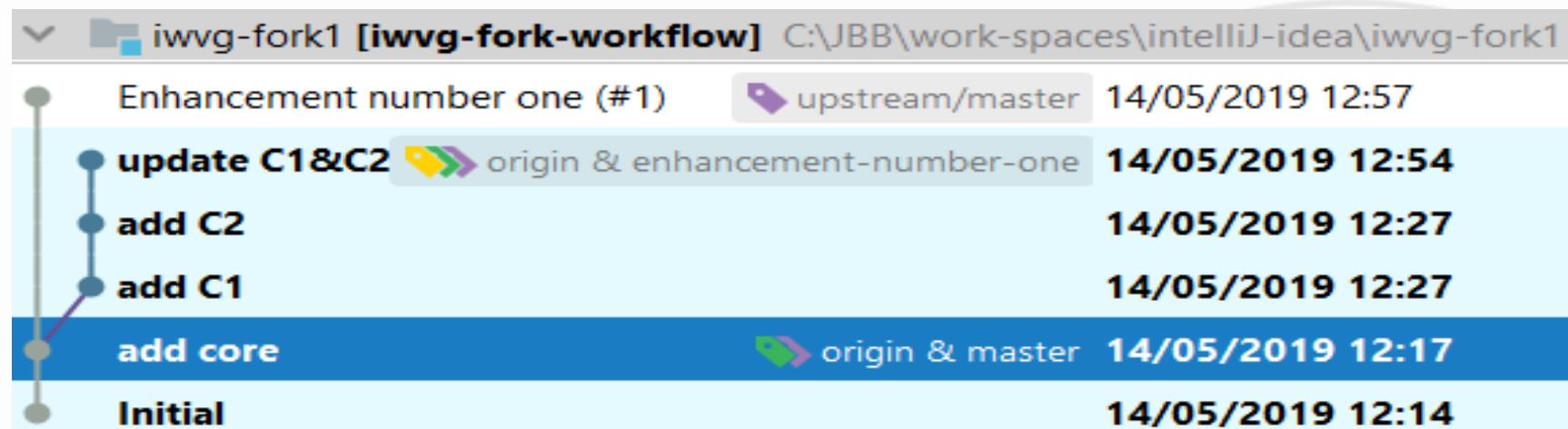
Notifications Unsubscribe

You're receiving notifications because you authored the thread.

2 participants MIW

GitHub

Flujo de Trabajo Bifurcación



Git. Forking Workflow



Los alumnos deben hacer fork al repositorio *iwvg-fork-workflow*

Se debe crear una clase con un método y realizar un *pull request*

Observar como evoluciona el código

Plan



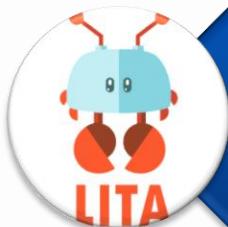
GitHub



Creately



Slack



Lita

The screenshot shows a GitHub repository page for 'miw-upm / iwg-ecosystem'. The top navigation bar includes 'Marketplace' and 'Explore' links. Below the repository name, there are links for 'Code', 'Issues 9', 'Pull requests 0', 'Projects 1', 'Wiki' (which is highlighted with an orange underline), 'Insights', and 'Settings'. The 'Wiki' section contains the following content:

Markdown

Máster en Ingeniería Web edited this page 2 days ago · 13 revisions

IWVG. Ecosistema. wiki!

Lenguaje Markdown

Varios

Línea en blanco para separar estructuras

\# carácter de escape para los caracteres especiales de Markdown

Wiki

Una es un sitio web cuyas páginas pueden ser editadas directamente desde el navegador, donde los usuarios crean, modifican o eliminan contenidos compartidos.

GitHub dispone de Wiki para la generación de documentación rápida y compartida a partir de lenguajes de marcado ligeros... *Markdown*.

GitHub Hitos

The screenshot shows the GitHub interface for the repository 'miw-upm/iwvg-ecosystem'. The 'Milestones' tab is selected. Two milestones are listed:

- Curso 2018-19**: Progress bar at 66% complete, 1 open ticket, 2 closed tickets. Last updated 2 days ago.
- Curso 2017-18**: Progress bar at 100% complete, 0 open tickets, 5 closed tickets. Last updated 2 days ago.

At the bottom of the screenshot, the word 'Milestone' is displayed in large blue text.

Representa una de fecha de referencia, normalmente asociada a un lanzamiento o finalización de un módulo.

Puede estar abierto-cerrado.

Se les puede asociar tickets.

Tiene marcado un nivel de finalización, obtenido por los tickets asociados que se encuentran cerrados.

GitHub Issues

The screenshot shows a GitHub Issues page for the repository "miw-upm / iwvg-devops". The top navigation bar includes links for Code, Issues (11), Pull requests (0), Actions, Projects (1), Wiki, Security, Insights, and Settings. Below the navigation is a search bar with the query "is:issue is:open". The main area displays two sections: "Open" and "Closed".

Open Issues:

- una points: 5 type: enhancement
#29 opened on 14 Sep 2019 by miw-upm
- Git 11 error type: documentation
#26 opened on 12 May 2019 by miw-upm
- Story 2 points: 8 priority: low type: test
#14 opened on 10 May 2019 by miw-upm
- Story 3 points: 3 priority: medium type: enhancement
#13 opened on 10 May 2019 by miw-upm. Labels: Scrum: sprint 1

Closed Issues:

- Story 1 2h 45m points: 2 priority: high type: enhancement
#15 by miw-upm was closed on 10 May 2019. Labels: Scrum: sprint 1
- Nota 2h points: 1 priority: high type: enhancement
#6 by miw-upm was closed on 10 May 2019. Labels: Curso 2018-19

Issues

Disputa, asunto, tarea, error, ticket...

Se abre un foro de comentarios.

Etiquetas: facilitan su identificación y organización.

Etiquetas de prioridad: priority: high, priority: médium...

Etiquetas de tipo: type: test, type: documentation...

Etiquetas de estimación: points: 8

GitHub

Organización de un Issue

The screenshot shows a GitHub issue page for repository 'Git 11' with issue number #26. The issue is open and was created by miw-upm 2 hours ago. It has 0 comments. The issue title is 'Ejercicios básicos de GIT. Situaciones anómalas (@miw-upm)' and includes a link to 'Más información...'. The issue has two labels: 'error' (red) and 'type: documentation' (grey). The 'Assignees' section shows 'miw-upm' assigned. The 'Labels' section shows 'error' and 'type: documentation'. The 'Projects' section indicates 'None yet'. The 'Milestone' section indicates 'No milestone'. The 'Notifications' section shows '1 participant' and an 'Unsubscribe' button. The commit history shows three commits from miw-upm referencing this issue: 'add C2:m9 #26', 'Merge issue #26 into develop', and another commit that is partially visible. The bottom of the page features a rich text editor with 'Write' and 'Preview' buttons, and a comment input field with placeholder 'Leave a comment'.

Issue

Breve descripción, en la wiki se detalla. Con la referencia en los commits #xx aparece asociado al issue#

GitHub Scrum

The screenshot shows a GitHub repository named "iwvg-ecosystem" with a Scrum board view. The repository has 1 issue, 2 pull requests, 1 project, and 14 forks. The board has three columns: Backlog, In progress, and Done. Each column has a "Manage" button at the bottom.

Story	Backlog	In progress	Done
#1 Story 5 #11 opened by miw-upm points: 5 priority: low type: enhancement	#3 Story 3 #13 opened by miw-upm points: 3 priority: medium type: enhancement	#6 Story 6 #10 opened by miw-upm points: 2 priority: high type: enhancement	#1 Story 1 #15 opened by miw-upm 2h 45m points: 2 priority: high type: enhancement
#2 Story 2 #14 opened by miw-upm points: 8 priority: low type: test	#4 Story 4 #12 opened by miw-upm points: 2 priority: medium type: bug		

Scrum

Aproximación a Scrum: es un proyecto *Automated Kanban*. Los hitos representan un Sprint. Con la etiquetas de point, se establece la estimación ...

Story

- Crear las historias (*issues*) en el proyecto. Asociarle *prioridad, estimación, tipo...*

Backlog

- Mover las historias a la columna *backlog*, son las que desarrollan en el próximo *sprint*, asociarles el *sprint* (hito).

Sprint

- Cuando alguien inicia una historia, tarea (*issue*), se lo asigna y lo mueve a la columna *In progress*.

Branch: issue#

- Se crea la rama 'issue#xx' y se programa la historia, tarea... un *commit* debe ser reflejado en el historial del *issue*, se añade en el mensaje '#xx'

Fusión

- Cuando la historia se termina, se fusiona con *develop*:
• git merge -no-ff -m "Merge #xx into develop. Detalles" issue#xx

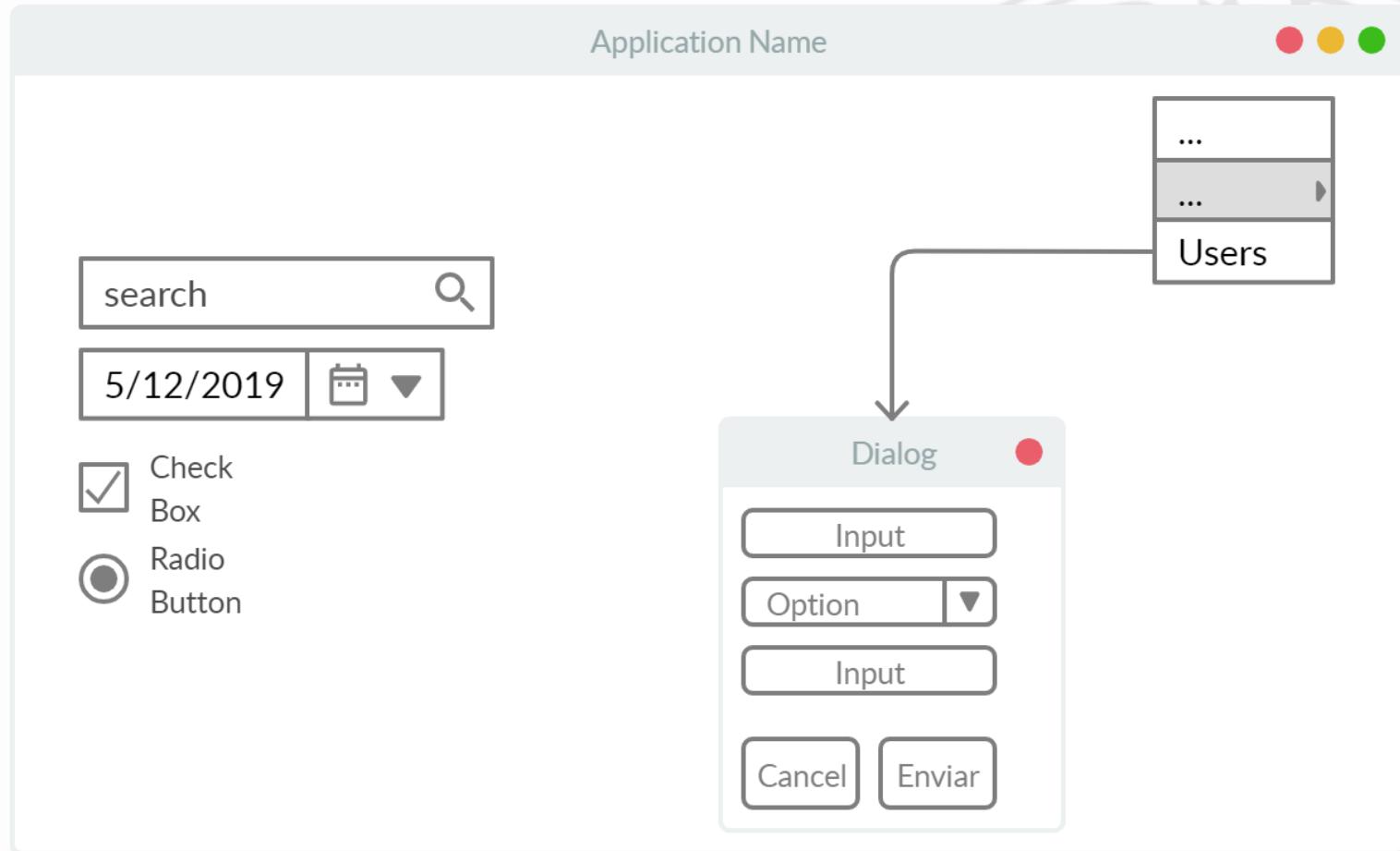
Close

- Se cierra la historia (*issue#xx*), se establece el tiempo consumido y se abandona la rama

Workflow exercises with Scrum: Note 3

- Proyecto: <https://github.com/miw-upm/iwvg-git-workflow/projects>

Creatily Mockups



Slack

Comunicación

The screenshot shows the Slack web interface for the 'miw-upm' workspace. The left sidebar includes sections for 'Primeros pasos', 'Hilos de conversaciones', 'Borradores', 'Personas', 'Aplicaciones', 'Archivos', and 'Canales'. The 'Canales' section highlights the '# devops' channel, which is selected and shown in the main pane. The main pane displays messages from various sources: 'Travis CI' (Build #250 and #251), 'GitHub' (Successful deployment of commit b9cbd76), and 'miw' (Adjunto PDF and Adjunto JAVA). A code snippet of a Java Lambda class is shown in the message from 'miw'. At the bottom, there is a message input field with placeholder text 'Enviar mensaje a #devops'.

Slack | devops | miw-upm

app.slack.com/client

#devops

Travis CI APP 17:21

Build #250 (b9cbd76) of miw-upm/lwvg-devops@develop by jbernal passed in 2 min 17 sec

Miércoles, 15 de abril

Build #251 (b9cbd76) of miw-upm/lwvg-devops@master by jbernal passed in 3 min 1 sec

GitHub APP 17:23

miw-upm

Successfully deployed b9cbd76 to lwvg-devops

miw-upm/lwvg-devops

Hoy

miw 12:00

Adjunto PDF

miw. Exámenes 2019-2020.pdf

miw 12:13

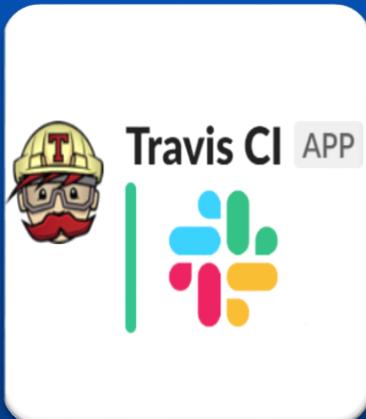
Adjunto JAVA

Lambda.java

```
1 public class Lambda {  
2     public static final Consumer<String> logInfoDetail = // accept(T)  
3         msg -> LogManager.getLogger(Lambda.class).info("Consumer: {}", msg);  
4  
5     public static final Function<String, Integer> convertToInt = // apply(T): R
```

Enviar mensaje a #devops

Slack Integración



Travis-CI → Slack

- Añadir Aplicación *Travis-CI* en *Slack*
- Instalar *ruby* (*>ruby -v*)
- Instalar el *CLI* de *Travis-CI*: *>gem install travis* (*>travis version*)
- Encriptar *Token* mediante *CLI* de *Travis-CI*
 - *>travis encrypt "miw-upm:token#devops" --add notifications.slack*



GitHub → Slack

- Se añade la aplicación de *GitHub*
- Escribir en Slack: */github help*, para ver los comandos
- Se debe conectar con la GitHub siguiendo los enlaces...
- En el canal *#devops*, se activa subscripción escribiendo:
 - */github subscribe miw-upm/iwvg-devops issues,commits,releases*

¿Qué es la Integración Continua?

Continuous Integration

Práctica del desarrollo del software para integrar código frecuentemente

El objetivo es mejorar la productividad y mantener la calidad del código.

Debe ser un proceso automático tras la detección del cambio del código en el repositorio para encontrar y arreglar los errores con rapidez .

Práctica relacionada con las *metodologías ligeras* y especialmente con XP. Evita integraciones infernales. Fue propuesto inicialmente por *Martin Fowler*.

Principios

Contribuir a menudo: DIA

No contribuir con código roto

Solucionar los test rotos inmediatamente

Mantener la calidad

No continuar con código roto



Travis CI

- Es un servicio de integración continua gratuito en la nube integrado con GitHub y ofrece soporte para: Java, Phython, Node.js, PHP, Ruby...



Jenkins

- Jenkins es un software de integración continua de código abierto escrito en Java.

Integración GitHub-TravisCI

.travis.yml

```
language: java
branches:
  only:
    - develop
    - /^release-[0-999].[0-999]$/i
    - master
notifications:
  email:
    recipients:
      - j.bernal@upm.es
script:
  # Unit Test & Integrations Test
  - mvn verify
```

Configuración

Se configura con el fichero *.travis.yml* en la raíz del proyecto.

Activar en *GitHub* el Servicio *Webhooks*, se realiza desde la Web de *Travis-CI*.

Travis CI

The screenshot shows the Travis CI interface for the repository `miw-upm/iwvg-ecosystem`. The build status is **passing**.

The build history table lists three successful builds for the `develop` branch:

Build	Commit	Time	Actions
#112	d380985	about 4 hours ago	Merge issue #26 into develop
#111	3c328d4	about 4 hours ago	Merge issue #25 into develop
#110	5cc28cc	about 4 hours ago	Merge issue #24 into develop

Each row includes a green checkmark icon, the build number, the commit hash, the time since the build, the merge issue number, and the target branch.

¿Qué es el Análisis Estático?

Análisis Estático

Es un análisis del código sin ejecución.

Nos ayudan a detectar problemas del código fuente, buscando errores de seguridad, código duplicado, problemas potenciales por mala práctica...

Análisis Estático



Sonarcloud

- Se ofrece en la nube y se integra
GitHub & Travis-CI



BetterCodeHub

- Se ofrece en la nube y se integra
con *GitHub*

Análisis Estático

Sonarcloud

Crear cuenta en Sonarcloud

Crear una nueva organización

Generar ApiKey

Guardar ApiKey en Travis-CI

Integrar con Travis: .travis.yml

- Debéis crear una organización, y para ello es mas fácil en la pestaña **Import from GitHub**, crearla mediante el botón: **Choose one of your GitHub**.

- [https://sonarcloud.io/account/security/.](https://sonarcloud.io/account/security/)

- Crear variable de entorno en Travis-CI: SONAR.
- Otra opción es encriptarla mediante *Travis-CLI*

- mvn sonar:sonar
 - Dsonar.host.url=https://sonarcloud.io
 - Dsonar.organization=*** -Dsonar.login=\$SONAR

Análisis Estático

Sonarcloud

Máster en Ingeniería Web / es.upm.miw.iwvg-devops master +

Overview Issues Measures Code Activity Administration ▾

Quality Gate Passed

Reliability [Measures](#)

 0 A
Bugs

started 10 days ago

Security [Measures](#)

 0 A
Vulnerabilities

 2 A
Security Hotspots

Maintainability [Measures](#)

 2h A
Debt

 21 A
Code Smells

Coverage [Measures](#)

 55.4%
Coverage

 14 A
Unit Tests

Duplications [Measures](#)

 10.3%
Duplications

 2 A
Duplicated Blocks

Análisis Estático

Better Code Hub

QA

- Verifica que su código cumpla con 10 pautas de ingeniería de software de referencia.

Lenguajes

- Soporta Java, Kotlin, TypeScript, Python...

Referencias

- Building Maintainable Software, Java Edition. Ten Guidelines for Future-Proof Code.
- Building Software Teams: Ten Best Practices for Effective Software Development.

Integración

- GitHub
- Configuración: `.bettercodehub.yml` en la raíz del proyecto.

Análisis Estático

Better Code Hub

Write Short Units of Code (*method*)

- Limit the length of code units to 15 lines of code.
- Do this by splitting long units into multiple smaller units until each unit has at most 15 lines of code.

Write Simple Units of Code (*method*).

- Limit the number of branch points per unit to 4.
- Do this by splitting complex units into simpler ones and avoiding complex units altogether.

Write Code Once.

- Do not copy code.
- Do this by writing reusable, generic code and/or calling existing methods instead.

Keep Unit Interfaces Small (*parameters*).

- Limit the number of parameters per unit to at most 4.
- Do this by extracting parameters into objects.

Separate Concerns in Modules (*class*).

- Avoid large modules in order to achieve loose coupling between them.
- Do this by assigning responsibilities to separate modules and hiding implementation details behind interfaces.

Análisis Estático

Better Code Hub

Couple Architecture Components Loosely *class-package*

- Achieve loose coupling between top-level components.
- Do this by minimizing the relative amount of code within modules that is exposed to (i.e., can receive calls from) modules in other components.

Keep Architecture Components Balanced *class-package*

- Balance the number and relative size of top-level components in your code.
- Do this by organizing source code in a way that the number of components is close to 9 (i.e., between 6 and 12) and that the components are of approximately equal size.

Keep Your Codebase Small (*project < 175.000 lines*)

- Keep your codebase as small as feasible.
- Do this by avoiding codebase growth and actively reducing system size.

Automate Tests

- Automate tests for your codebase (>80%).
- Do this by writing automated tests using a test framework.

Write Clean Code

- Write clean code.
- Do this by not leaving code smells behind after development work.

Análisis Estático

Better Code Hub

Compliance **9** of 10

miw-upm/
apaw-microservice-themes-user

Last analysis: 3 days ago
Previous analysis: 3 days ago

Branch: develop (default)

	Write Short Units of Code	✓	...
	Write Simple Units of Code	✓	...
	Write Code Once	✗	...

Refactoring candidates

	Show snoozed
<input checked="" type="checkbox"/> Duplicate	Lines of Code
<input type="checkbox"/> 12 lines occurring 2 times in 2 files: Address.java, UserCreationDto.java	12
<input type="checkbox"/> 8 lines occurring 2 times in 2 files: User.java, UserCreationDto.java	8

non-duplicated code duplicated code

	Keep Unit Interfaces Small	✓	...
	Separate Concerns in Modules	✓	...
	Couple Architecture Components Loosely	✓	...
	Keep Architecture Components Balanced	↑ ✓	...
	Keep Your Codebase Small	✓	...
	Automate Tests	✓	...
	Write Clean Code	✓	...

CI

{→}

CI con Travis-CI, Sonarcloud & BetterCodeHub

- ...

¿Qué es la Entrega Continua?

Continuous Delivery

Práctica del desarrollo del software para liberar código en ciclos cortos

El objetivo es que las mejoras del desarrollo sean visibles para el cliente lo antes posible y poder tener una retroalimentación.

Despliegue Continuo (*Continuous Deployment*) resulta cuando las entregas se despliegan automáticamente.

Servidores
Físicos

Servidores
Virtuales

La Nube

IaaS

Infraestructura
como Servicio

Se ofrece
almacenamiento,
redes y servidores

PaaS

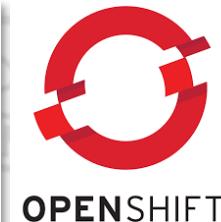
Plataforma como
Servicio

Se ofrece un entorno
para distribuir
aplicaciones.

SaaS

Software como
Servicio

Se ofrecen
aplicaciones





Plataforma como Servicio
(*PaaS*)

Comercializada por
Salesforce

Soporta muchos lenguajes:
Java, NodeJS, Python, PHP...

Heroku

- Tiene una muy alta escalabilidad: procesos *dyno* dinámicos bajo demanda.
- Cada instancia (*dyno*) contiene una copia del código fuente de la aplicación.
- Es la plataforma Heroku la que se ocupa de colocar un servidor web delante de los *dyno*, de hacer las comunicaciones SSL, etc.
- Los procesos *dyno* pueden tener una vida muy efímera (por ejemplo, para atender tan solo una petición)
- No es recomendable almacenar información en memoria, ni en ficheros.

Crear aplicación

- *New>Create new app*

Configurar

- Automático

Obtener API-key

- *Personal>Account settings*

Integrar con Travis

- Añadir el comando *deploy* en el fichero *.travis.yml*

Travis-CI

- En *Travis*, crear la variable de entorno *HEROKU* con la ApiKey.

Code

- Crear el fichero *Procfile*, en la raíz del proyecto.

Heroku CLI

- Es un cliente de consola

```
C:\work-spaces\depops>heroku -versión
C:\work-spaces\depops>heroku login
C:\work-spaces\depops>heroku logs --app=<proyecto> -n 100
C:\work-spaces\depops>heroku logs --tail -app=<proyecto>
```

CD con Heroku

• . . .