

<b>CS</b> Canigó	MANUAL D'USUARI SIC	0192
	SIC Manual Usuari.docx	
	N. versió: 3.0.2	Pàg. 1 / 26

Servei 8.30 – Arquitectura de Desenvolupament

# Manual d'Usuari


## Servei d'Integració Continua

**Tema/Detall:**

Manual d'usuari del Servei d'Integració Continua.


Versió	Data	Autor	Comentaris
3.0.0	05/05/2017	CS Canigó	Autoservei de repositoris i usuaris
3.0.1	04/07/2017	CS Canigó	Gestió binaris + Jobs desplegament CPD
3.0.2	04/12/2017	CS Canigó	Adaptacions de Jobs desplegament CPD
Aprovació		Data	Signatura

	Preparat	Revisat	Aprovat	Autoritzar
Nom				
Signatura				
Data				


	MANUAL D'USUARI SIC	0192
	SIC Manual Usuari.docx	
	N. versió: 3.0.2	Pàg. 2 / 26

## Índex

1 Introducció.....	4
1.1 Objecte.....	4
1.2 A qui va dirigit.....	4
1.3 Abast.....	4
2 GLOSSARI DE TERMES.....	5
3 Publicació del codi font al GitLab.....	7
3.1 Actualització del codi font al repositori local.....	7
3.2 Publicació del codi font al repositori de GitLab.....	7
3.3 Normativa d'ús de GitLab.....	7
4 Gestió de binaris.....	9
4.1 Pujada de binaris.....	9
4.2 Baixada de binaris.....	10
5 Ús de Jenkins.....	11
5.1 Accés.....	11
5.2 Visualització de Resultats.....	13
5.3 Funcionament dels jobs Pipeline.....	15
5.3.1 Execució del job i consideracions prèvies.....	15
5.3.2 Etapes dels jobs Pipeline.....	15
5.3.3 Stage INIT.....	16
5.3.4 Stage CHECKOUT.....	16
5.3.5 Stage BUILD.....	16
5.3.6 Stage COMMIT TEST.....	16
5.3.7 Stage UNIT TEST.....	16
5.3.8 Stage ANÀLISI ESTÀTIC de CODI.....	17
5.3.9 Stage Generació tag de Build.....	17
5.3.10 Stage INT.....	17
5.3.11 Stage SMOKE TEST.....	17
5.3.12 Stage PRE.....	17
5.3.13 Stage SMOKE TEST.....	18
5.3.14 Stage ACCEPTANCY TEST.....	18
5.3.15 Stage EXPLORATORY TEST.....	18
5.3.16 Stage GENERACIÓ TAG DEFINITIU.....	18
5.3.17 Stage PRO.....	19
5.3.18 Stage SMOKE TEST.....	19

	MANUAL D'USUARI SIC	0192
	SIC Manual Usuari.docx	
	N. versió: 3.0.2	Pàg. 3 / 26

5.3.19 Resultats del job i arxivat d'artefactes.....	20
5.4 Jobs desplegament automàtic per a CPD.....	21
5.5 Execució de scripts de BBDD durant els desplegaments.....	23
5.6 Instal·lació de llibreries JEE, Microsoft i mòduls npm i bower.....	26
5.6.1 Introducció.....	26
5.6.2 Execució del job.....	26
6 Format de l'arxiu sic.yml.....	26

	MANUAL D'USUARI SIC	0192
	SIC Manual Usuari.docx	
	N. versió: 3.0.2	Pàg. 4 / 26

## 1 Introducció

### 1.1 Objecte

L'objectiu d'aquest document és oferir als futurs usuaris del SIC un manual de funcionament de la plataforma. En aquest manual es descriuen tots els flux d'execució possibles així com la interacció que tenen amb els usuaris.

Aquest manual agafa com a punt de partida el fet que l'usuari estigui donat d'alta en la plataforma i que l'aplicació o aplicacions amb les que vulguin interaccionar també estiguin donades d'alta.

L'estructura d'aquest document està organitzada en torn a les tasques que els usuaris poden realitzar dintre de la plataforma. S'han previst els següents cassos d'ús:


- Instal·lació i ús del client de GIT per accedir al repositori de codi.
- Autenticació sobre la plataforma.
- Accés als diferents Jobs.
- Visualització genèrica de resultats.

### 1.2 A qui va dirigit

Aquest document va dirigit a tots els usuaris del Servei d'Integració Contínua.

### 1.3 Abast

L'abast d'aquest document inclou els processos propis del SIC des de la perspectiva dels release managers i responsables de projecte. Existeixen una sèrie de requisits pel que fa al software, queda però fora de l'abast d'aquest document detallar-los. Es troben ben definits en la "Norma J2EE dels serveis TIC Centrals" i en la "Guia de Desenvolupament en J2EE" per aplicacions JAVA, i els documents "Nomenclatura i esquema de base de dades" (norma SC-NOR11-01) i Desenvolupament WEB (NOR27). Cal que els desenvolupadors els coneguin be per assegurar que codi entregat pugui ser desplegat als Serveis TIC Centrals.

	MANUAL D'USUARI SIC	0192
	SIC Manual Usuari.docx	
	N. versió: 3.0.2	Pàg. 5 / 26

## 2 GLOSSARI DE TERMES

- **Integració Continua (CI)**

La integració contínua (o CI, per Continuous Integration) és una pràctica de desenvolupament de programari en la que els membres d'un equip integren la seva feina freqüentment, en general una vegada com a mínim al dia – generant múltiples integracions al dia. Cada integració és verificada per una construcció automàtica (incloent les proves) per detectar errors d'integració tan ràpid com sigui possible.

- **Servei d'Integració Continua (SIC)**

El Servei del CTTI que dóna suport a la integració continua, així com a d'altres aspectes del cicle de vida del programari

- **Cicle de vida del programari**

El cicle de vida del programari és el concepte que engloba tots els aspectes lligats al desenvolupament d'una aplicació, des de la presa de requeriments fins al desplegament en un entorn productiu. També conegut com a ALM (per Application Lifecycle Management).

- **Jenkins**

Jenkins és una eina d'integració continua 100% Java que s'executa en un servidor d'aplicacions tipus Tomcat o WebLogic. És un projecte de programari lliure que darrerament ha obtingut molta popularitat i diversos premis. El SIC està basat en Jenkins. Jenkins necessita tenir per sota una eina de gestió de la construcció i un repositori de codi.

- **Sistema de Gestió de la Construcció**


Un Sistema de Gestió de la Construcció (o Build Management, o Build Automation System) permet automatitzar en un script les tasques pròpies del desenvolupament diari com: generació d'executables a partir del codi font, execució de proves unitàries, creació de documentació (javadoc), etc.

- **Maven**

Maven és una eina de programari lliure que permet l'automatització de la construcció. A partir del codi font i un fitxer descriptor del projecte és capaç de generar els executables a desplegar. Una de les seves grans virtuts és la gestió de les dependències del projecte (de quines llibreries depèn, i de quina versió d'elles). Té una arquitectura basada en plug-ins que li permeten fer moltes més coses.

- **Repositori de codi / Sistema de Control de Versions**

Un sistema de control de versions manté diferents versions de molts tipus de documents i permet marcar-los (**tags**) i seguir múltiples camins d'evolució (**branches**) a partir d'una branca per defecte (**master**). El lloc on físicament s'emmagatzemen totes les versions és

	MANUAL D'USUARI SIC	0192
	SIC Manual Usuari.docx	
	N. versió: 3.0.2	Pàg. 6 / 26

el repositori. El procés de copiar una nova versió d'un fitxer en el sistema de control de versions local s'anomena **commit** o **check-in**. La publicació d'aquesta nova versió al repositori central del SIC s'anomena **push**. En el cas d'un entorn de desenvolupament, això permet disposar d'un conjunt coherent de fitxers de codi font a partir dels quals generar els executables, així com mantenir múltiples bases de codi per separat (per exemple, la versió 1.x i la versió 2.x que evolucionin en paral·lel i per separat).

- **Git**

És un sistema de control de versions pensat especialment per aplicacions amb molt arxius de codi font i gestionades per molta gent.

Característiques principals:

- Rapidesa en la gestió de branques i barreja de versions: molt potent per desenvolupaments no lineals.
- Gestió distribuïda: cada desenvolupador té una còpia local del codi sencer.

- **GitLab**

És un aplicatiu que publica una interfície web per facilitar l'administració i l'ús del sistema de control de versions GIT.

- **Jenkins Pipeline**

Una Pipeline de Jenkins és un nou tipus de job de Jenkins disponible en el core del producte des de la versió 2.0. Aquest nou tipus de job permet tenir la configuració del mateix en un arxiu anomenat jenkinsfile. El jenkinsfile és un arxiu groovy que recull les tasques que s'han d'executar i permet dividir-les en *stages* (fases).


- **Release Manager / Gestor de Lliuraments**

Persona de l'equip de desenvolupament que s'encarrega gestionar les versions d'una aplicació. És qui efectua els commits i posa els tags de versió en el repositori de codi.

- **Desplegament Automatitzat**

L'automatització del desplegament consisteix en escriure un script que contingui les tasques relacionades amb el desplegament d'una versió d'una aplicació, fent servir les interfícies proporcionades pels sistemes on s'ha de desplegar. Això permet reduir els punts de fallida ja que s'automatitzen tasques repetitives. Els servidors d'aplicacions com Weblogic acostumen a donar una API que pot ser invocada des d'una eina de construcció com Maven. El SIC fa ús d'aquestes funcionalitats per a assolir desplegaments automàtics als entorns d'integració.

En el cas de les tecnologies Microsoft, els servidors IIS poden permetre la connexió, la instal·lació y l'actualització d'aplicacions des d'un equip client utilitzant MS Web Deploy.

	MANUAL D'USUARI SIC	0192
	SIC Manual Usuari.docx	
	N. versió: 3.0.2	Pàg. 7 / 26

### 3 Publicació del codi font al GitLab

#### 3.1 Actualització del codi font al repositori local

Els desenvolupadors hauran de tenir instal·lat/configurat el SCM Git al seu equip.

Git proporciona a cada desenvolupador una còpia local del codi font de l'aplicació (repositori local). El desenvolupador anirà actualitzant aquest "repositori local" amb els seus canvis executant COMMIT.

#### 3.2 Publicació del codi font al repositori de GitLab

Una vegada s'hagi validat internament el codi de l'aplicació, es publicarà al repositori GitLab executant PUSH.

En aquest moment, al fer el PUSH al GitLab, automàticament es llençarà al Jenkins la execució del Pipeline associat a l'aplicació, en el cas que l'aplicació tingui un job creat a tal efecte (no s'habilitarà per als Release Manager l'execució manual del job).

Les múltiples tasques que fa aquest Pipeline, es descriuen a l'apartat 4.

#### 3.3 Normativa d'ús de GitLab

##### 3.3.1 Estructuració de repositoris


Al GitLab del SIC, es troben pre-creats una sèrie de grups que es corresponen -un a un- amb tots els codis d'aplicació. D'aquesta manera, cada grup disposarà de tot el codi font corresponent al codi d'aplicació que ve representat pel seu nom.

Dins de cadascun d'aquests grups, s'albergaran tots els projectes del codi d'aplicació. Hi haurà codis d'aplicació amb només un sol projecte i n'hi haurà que tindran més d'un. El criteri general per decidir què és un projecte i que no és preguntar-se si aquest conjunt de codi font és susceptible de ser versionat de forma independent al de la resta de projectes del codi d'aplicació.

Per exemple, una aplicació basada en microserveis, requerirà un projecte per a cada microservei.

##### 3.3.2 Limitacions

Per a propiciar el bon ús del repositori i les bones pràctiques en la gestió del cicle de vida de les aplicacions, s'han establert les següents limitacions:

	MANUAL D'USUARI SIC	0192
	SIC Manual Usuari.docx	
	N. versió: 3.0.2	Pàg. 8 / 26

1. En quant a la mida:

1. La mida màxima dels arxius serà de 20 MB. Qualsevol arxiu multimèdia hauria més gran que aquesta mida màxima hauria de proporcionar-se mitjançant una altra via.

2. En quant al nom:

1. No es permet l'existència de carpetes amb nom «node\_modules». Aquesta carpeta es sol utilitzar per a la descàrrega de dependències en aplicacions Node.js en el procés de construcció. Per tant, no cal afegir-la al repositori, ja que el propi Jenkins l'obtindrà en el procés de construcció.

3. En quant a l'extensió:

1. No es permet l'existència d'arxius amb les següents extensions:

- JAR
- WAR
- EAR
- DLL
- EXE

Aquestes extensions són pròpies d'arxius binaris. Per a la compartició de binaris amb CPD s'ha habilitat un sistema alternatiu a l'actual SVN. Aquest sistema resta explicat a l'apartat *"4. Gestió de binaris"*.

Si aquests binaris són llibreries generades per terceres parts, s'haurien de depositar al repositori d'artefactes Nexus (demanar-ho via petició de Suport a "Framework SIC" via Remedy ).


Tota violació de les limitacions aquí exposades invalidaran el push al servidor oficial. Durant el procés de push, es mostrarà a l'usuari un missatge d'error amb els incumpliments detectats.

### 3.3.3 Carpeta especial /sic

Es requereix l'existència de la carpeta especial **/sic**. Aquesta carpeta albergarà el següent contingut:

- L'arxiu **sic.yml**, que serà un arxiu que inclourà, entre altres dades, la següent informació:
  - La versió de l'aplicatiu que s'està pujant. Haurà de seguir la codificació requerida per qualitat (<versióMajor.versióMenor.versióFix). Per exemple: 1.0.1.
  - Al capítol 6 podreu trobar més informació respecte al format que ha de tenir aquest arxiu.



	MANUAL D'USUARI SIC	0192
	SIC Manual Usuari.docx	
	N. versió: 3.0.2	Pàg. 9 / 26

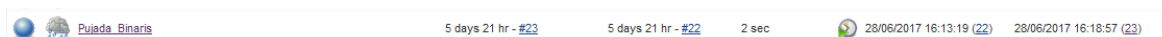
## 4 Gestió de binaris

Actualment al SIC els binaris es depositen al repositori de codi SVN. Degut a la pròpia naturalesa dels sistemes de gestió de codi font, els elements introduïts en aquest tipus de sistemes mai poden eliminar-se, provocant multitud de problemes en la seva gestió.

Per donar solució a aquest problema, s'ha dotat al SIC d'un nou espai per emmagatzemar binaris. Els proveïdors d'aplicacions podran pujar aquests arxius mitjançant un job de Jenkins i tant ells mateixos com el CPD/LLT que s'encarrega de desplegar-los podran accedir-hi en mode lectura a través d'un frontal web.

### 4.1 Pujada de binaris

En accedir a la Plataforma Jenkins, tots els usuaris Release Manager tindran visibilitat sobre el job 'Pujada\_binaris'.



Caldrà accedir a aquest picant sobre el seu nom.

Una vegada dins, executar-ho de manera anàloga a la resta de jots (opció de menú lateral '*Build with Parameters*').

Apareixerà un formulari on s'hauran d'introduir les dades de l'aplicació:

**Project Pujada\_Binaris**

This build requires parameters:

CODI\_APLICACIO  Obligatori

NOM\_APLICACIO  Obligatori


VERSIO  Obligatori

ARXIU\_ARTEFACTES  No se ha seleccionado ningún archivo. Obligatori (Fitxer de binaris a pujar)

ARXIU\_DOCUMENTACIO  No se ha seleccionado ningún archivo. Obligatori (Fitxer de documentació a pujar)

On:

- CODI\_APLICACIO: Codi assignat a l'aplicació. Ha de contenir 4 dígit, completant amb 0 per la banda esquerra si cal. Exemple: 492 -> 0192
- NOM\_APLICACIO: Nom de l'aplicació/mòdul/líbreria. No ha de contenir espais en blanc, ni caràcters estesos (accents, dièresis,...). S'aconsella introduir l'acrònim de l'aplicació. Exemple : Framework Canigó -> FWKCanigo
- VERSIO: Número de Versió del binari a pujar. Exemple: 1.0.7
- ARXIU\_ARTEFACTES: Mitjançant el botó 'Examinar' caldrà escollir el binari o paquet (zip)

	MANUAL D'USUARI SIC		0192
	SIC Manual Usuari.docx		
	N. versió: 3.0.2		Pàg. 10 / 26

amb els binaris + scripts BBDD a pujar.

- ARXIU\_DOCUMENTACIO: Mitjançant el botó 'Examinar' caldrà escollir el Document de instruccions de desplegament o el paquet (zip) amb tots els documents de instruccions de desplegaments.

Amb tot informat, cal continuar amb l'execució del job picant sobre el boto *'Build'*.

El job realitzarà validará:

- Que el codi d'aplicació existeixi com a aplicació donada d'alta a l'inventari d'aplicacions oficial del Pòrtic.
- Que l'usuari que ha executat el job sigui un Release Manager del Lot i àmbit que manté l'aplicació amb el codi indicat.

Si tot es troba en ordre, el job haurà pujat ambdós fitxers al servidors de binaris del SIC, sota una estructura de directoris condicionada per el codi, nom i versió de l'aplicació indicats.

## 4.2 Baixada de binaris

Per a la baixada de binaris, cal accedir al portal de binaris del SIC, mitjançant la URL:

En intentar accedir-hi, demanarà credencials. S'han d'emprar les credencials del compte GICAR.



En accedir al portal, caldrà picar sobre la imatge amb descripció 'Recuperar artefactes del SIC'

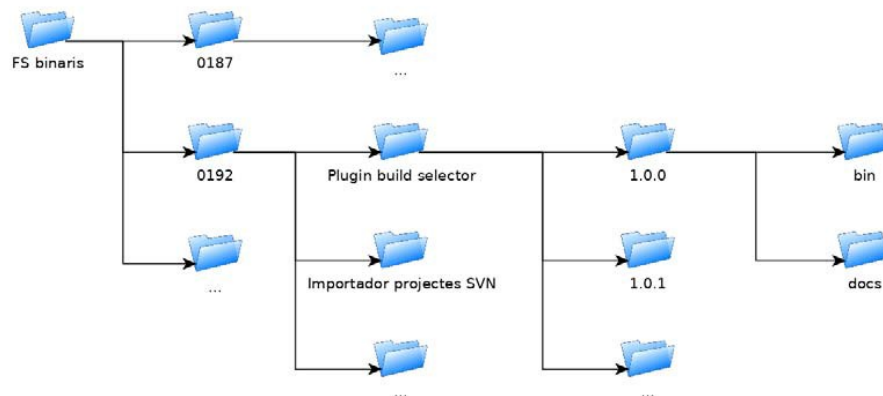
(també es pot accedir directament al context: )

Dins aquest, es veurà un directori Apache típic, on apareixerà un llistat de carpetes que es correspondran amb codis d'aplicació. Apareixeran només aquelles per a les que s'ha de tenir accés.

Dins cada carpeta, es trobaran subcarpetes amb el noms de les aplicacions.

Dins cada carpeta d'aplicació, es trobaran subcarpetes amb identificadors de versió de l'aplicació.

Per últim, dins cada carpeta de versió, figurarà una carpeta per als binaris i una altra per a la documentació.



Si es coneix la URL d'accés final on es troba publicat el binari, es pot accedir directament a la descàrrega d'aquest.

## 5 Ús de Jenkins

### 5.1 Accés

Per poder efectuar aquesta tasca l'usuari ha d'accedir a la plataforma mitjançant el formulari d'autenticació de Jenkins. La url d'accés és .

#	Estat
1	Ocíos
2	Ocíos
3	Ocíos






	MANUAL D'USUARI SIC	0192
	SIC Manual Usuari.docx	
	N. versió: 3.0.2	Pàg. 12 / 26

Figura 3.2.2 – 1






Una vegada fet el login, s'accedeix a la llista de tasques disponibles per l'usuari al menú de l'esquerra i a la vista central apareixerà una graella amb els jobs disponibles per l'usuari.

A la primera columna de la graella, apareixerà un semàfor indicant el resultat (**Status**) de la darrera execució del job:

Estat	Descripció
	El projecte encara no ha estat construït mai.
	L'última execució ha anat correctament.
	L'última execució ha anat correctament però és inestable.
	L'última execució ha fallat.

Taula 3.2.3 – 2


La segona columna es correspon a la salut general del Job (**Weather**). Es calcula la salut general del projecte basant-se en una sèrie d'indicadors. En el nostre cas es basaran en l'estabilitat, cobertura i tests.


Estat	Descripció
	Indica una salut d'entre 80-100%.
	Indica una salut d'entre 60-79%.
	Indica una salut d'entre 40-59%.
	Indica una salut d'entre 20-39%.
	Indica una salut d'entre 0-19%.

Taula 3.2.3 – 3

La resta de columnes venen descrites en la taula següent;

Columna	Descripció
---------	------------

	MANUAL D'USUARI SIC		0192
	SIC Manual Usuari.docx		
	N. versió: 3.0.2		Pàg. 13 / 26

Name	Nom del Job o Pipeline.
Darrer muntatge correcte	Temps des de l'última execució amb èxit del Job.
Darrer muntatge fallit	Temps des de l'última execució sense èxit del Job.
Darrera durada	Durada de l'última execució del Job
Last success Version 	Icona d'execució ràpida del Job. Llença una execució del Job (build).

Taula 3.2.3 – 4

## 5.2 Visualització de Resultats

De forma genèrica es pot consultar l'estat de finalització d'un Pipeline. Aquesta informació es pot visualitzar en la pàgina principal de cada Pipeline. Per poder accedir-hi a aquesta pàgina només s'ha de fer clic en el nom del job Pipeline (figura 3.2.3 – 1).

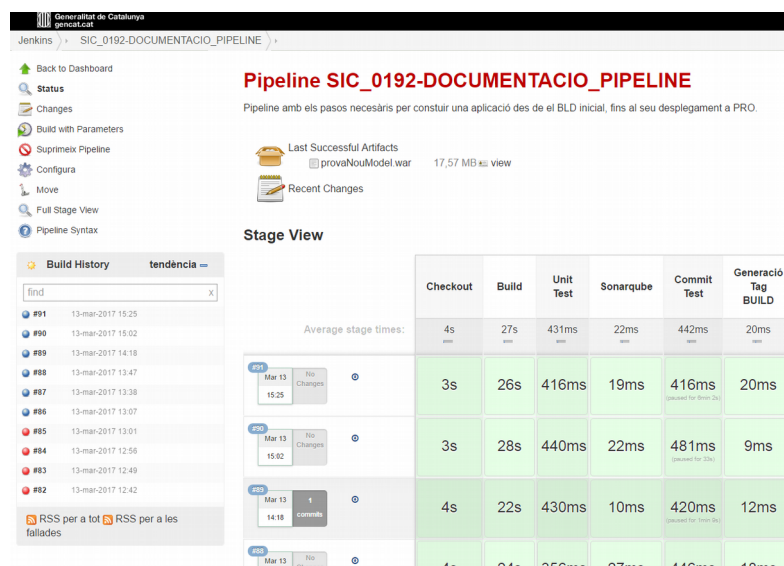


Figura 3.2.4 – 1

En el menú lateral esquerre es pot veure un quadre anomenat "Build History". Aquest quadre mostra amb una icona l'estat de salut general del projecte. El significat és consultable en la taula 3.2.3 – 3.

Llistat a continuació apareixen els últims builds d'aquest Job. L'estat de cada un d'ells ve representat per la icona que els precedeix (veure taula 3.2.3 – 2). Es disposa també d'unes

estadístiques del Job que poden veure's fent clic sobre l'enllaç "tendència".

### Build Time Trend

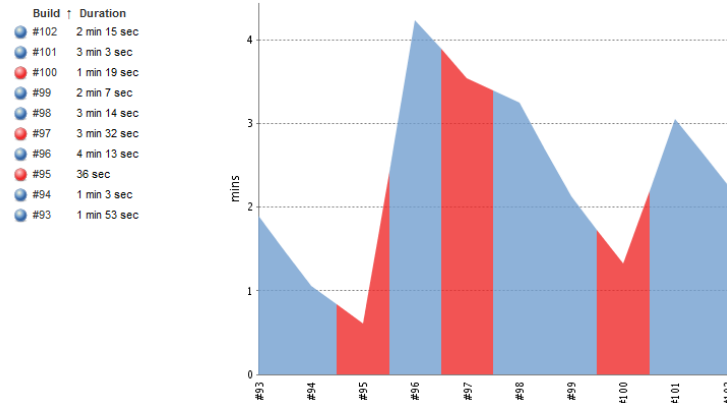


Figura 3.2.4 – 2


La gràfica mostra un històric dels builds executats sobre aquest Pipeline i el temps que han trigat cadascun així com el seu estat de finalització.

A la zona central de la pantalla es mostra una gràfica amb les darreres execucions del Pipeline, i el resultat a cadascuna de les etapes.

	Checkout	Build	Unit Test	Sonarqube	Commit Test	Generació Tag BUILD	INT	Smoke Test	Generació Tag DEFINITIU	PRE	Smoke Test	Acceptancy Test	Exploratory Test	PRO	Smoke Test
Average stage times:	4s	27s	431ms	22ms	442ms	20ms	31s	140ms	518ms	15s	NaN	NaN	18ms	17ms	16s
#95 Mar 13 15:25 No Changes	3s	26s	416ms	19ms	416ms (passed for 30s)	20ms	32s	24ms	491ms (passed for 10s)	18s	14ms	21ms	17ms	15s	484ms (50s)
#96 Mar 13 15:02 No Changes	3s	28s	440ms	22ms	481ms (passed for 30s)	9ms	29s	29ms	382ms (passed for 10s)	17s	12ms	15ms	13ms	15s	427ms (passed for 10s)
#95 Mar 13 14:18 1 console	4s	22s	430ms	10ms	420ms (passed for 10min 30s)	12ms	30s	23ms	575ms (passed for 170s)	18s	15ms	12ms	11ms	15s	444ms (passed for 170s)
#95 Mar 13 13:47 No Changes	4s	24s	356ms	27ms	446ms (passed for 30s)	18ms	33s	14ms	421ms (passed for 10s)	18s	23ms	23ms	21ms	16s	521ms (passed for 30s)
#97 Mar 13 13:38 No Changes	3s	23s	418ms	18ms	395ms (10s)	10ms	30s	9ms	553ms (passed for 20min 40s)	18s	20ms	19ms	22ms	16s	450ms (passed for 10s)
#96 Mar 13 13:07 No Changes	3s	26s	492ms	33ms	402ms (50s)	22ms	33s	19ms	527ms (passed for 10min 30s)	18s	33ms	20ms	19ms	18s	454ms (40s)
#95 Mar 13 13:01 No Changes	4s	25s	481ms	20ms	566ms (passed for 30s)	28ms	31s	20ms	880ms (passed for 10min 120s)	9s failed					

Si es vol més detall de l'estat d'un build d'un Job es pot fer clic sobre el build (figura 3.2.4 – 1). En fer això s'accedeix al detall de l'execució d'un build. La informació que es mostra depèn del tipus de Job que sigui. Tot i així sempre apareixerà la opció "Console Output". En fer clic sobre ella es podrà accedir al log de la tasca.

Al final d'aquest log es pot veure la paraula **SUCCESS** o **FAILED** que indica si el build va anar bé

	MANUAL D'USUARI SIC	0192
	SIC Manual Usuari.docx	
	N. versió: 3.0.2	Pàg. 15 / 26

o malament.

També pot donar-se el resultat **ABORTED**, el qual indicaria la cancel·lació del job per part de l'usuari.

En finalitzar el job o avortar-se per qualsevol problema, s'enviarà una notificació via mail al responsable del projecte.

## 5.3 Funcionament dels jobs Pipeline

### 5.3.1 Execució del job i consideracions prèvies


Els jobs tipus Pipeline no es podran invocar directament al portal Jenkins ni es podrà sol·licitar la seva execució a l'equip de SIC mitjançant una petició Remedy. Els jobs s'executaran quan es produeixi un push al projecte Git per part del lot d'aplicacions.

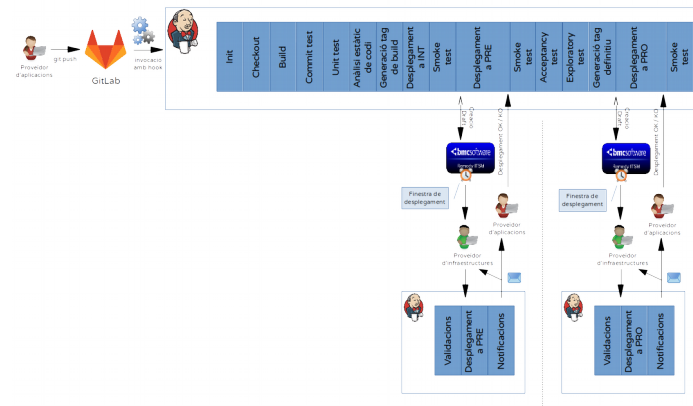
Es recomana ser curosos, ja que qualsevol push efectuat contra el repositori dispararà el job. Per tant, és important tenir en compte els següents punts abans fer les pujades al Gitlab de SIC:

1. Limitar la quantitat d'usuaris que utilitzin el servei Git del SIC. D'aquesta manera es té controlat més fàcilment quan i qui fa el push. Des de el SIC, sempre s'ha procurat que hi hagi la figura del Release Manager (gestor de revisions), que és l'encarregat entre d'altres tasques de pujar la versió final de l'entorn de desenvolupament al SIC.
2. Fer un únic push amb èxit per versió. És a dir, si el job falla en algun punt (construcció, desplegament, etc.) es pot tornar a fer un nou push amb les correccions pertinents al codi. Però un cop s'ha generat el TAG definitiu, no es permetrà fer el push de nou sense incloure una nova versió a l'arxiu del projecte **/sic/sic.yml**.

### 5.3.2 Etapes dels jobs Pipeline

Els jobs Pipeline realitzen multitud de tasques organitzades en STAGES. En cas de produir-se algun error a qualsevol etapa, l'execució del Pipeline es cancel·larà i s'enviarà un correu electrònic al responsable de l'aplicació informant del que ha passat.

	MANUAL D'USUARI SIC		0192
	SIC Manual Usuari.docx		
	N. versió: 3.0.2		Pàg. 16 / 26



Les etapes que contempnen els jobs Pipeline del SIC es descriuen a continuació.

### 5.3.3 Stage INIT

Aquesta etapa realitza una sèrie d'inicialitzacions necessàries per a l'execució del job pipeline.

### 5.3.4 Stage CHECKOUT

A aquesta etapa el Jenkins es connecta al repositori GitLab i es descarrega el codi font de l'aplicació, al seu workspace.

### 5.3.5 Stage BUILD

A aquesta etapa el Jenkins construeix l'aplicació i els artefactes pertinents a partir del codi descarregat a l'etapa anterior.

La construcció es fa d'acord a la naturalesa de l'aplicació, és a dir, per a aplicacions Java es fa mitjançant Maven, per a aplicacions .NET mitjançant MS Build, etc.

### 5.3.6 Stage COMMIT TEST


Aquesta etapa executarà els tests de commit, si s'escau. D'igual manera que a l'etapa de UNIT TEST, pot ser un requeriment haver-los de passar amb èxit.

Si els tests de commit s'executen amb èxit, es generarà un TAG, anomenat "versió construïble", al repositori de l'aplicació.

### 5.3.7 Stage UNIT TEST

A aquesta etapa s'executaran els tests unitaris, si s'escau. Pot ser un requeriment haver-ne de tenir i anirà en funció de l'aplicació.



	MANUAL D'USUARI SIC	0192
	SIC Manual Usuari.docx	
	N. versió: 3.0.2	Pàg. 17 / 26

Actualment, només es dóna suport a test unitaris JAVA a través de MAVEN.

### 5.3.8 Stage ANÀLISI ESTÀTIC de CODI

Aquesta etapa permetrà executar l'anàlisi de codi estàtic a través de Sonarqube. Actualment no està disponible i es fa un bypass a la següent etapa.

### 5.3.9 Stage Generació tag de Build

Aquesta etapa genera un tag de Build al repositori de codi. Aquesta tag significa que aquest commit correspon a una versió construïble. La nomenclatura dels TAG's serà:

```
<versio>.B001
<versio>.B002
...
<versio>.B00N
```

On <versio> és la versió indicada al fitxer /sic/sic.yml que proporciona l'aplicació.

Actualment, el SIC no executa tests de commit i, per tant, en aquesta etapa es genera el TAG descrit anteriorment.

### 5.3.10 Stage INT

Etapa que farà el desplegament automàtic de l'aplicació a l'entorn d'Integració. Cas que l'aplicació no compti amb aquest entorn, s'ometrà aquesta Stage.

### 5.3.11 Stage SMOKE TEST

Aquesta etapa realitzarà una validació bàsica per detectar que l'aplicació s'ha publicat correctament (com per exemple accedir a una URL de l'aplicació i veure que respon).

### 5.3.12 Stage PRE

En arribar a aquesta etapa, l'execució del job s'aturarà per demanar confirmació manual per a continuar amb la petició de desplegament a PRE.

Generació Tag BUILD	INT	Smoke Test	PRE	Smoke Test
13ms	1min 5s	74ms	4s	NaNy NaNd
Efectuar petició desplegament a PRE? <div> <input type="button" value="Proceed"/> <input type="button" value="Abort"/> </div>				(paused for 53s)

Per tal que aparegui la finestra modal demanant la confirmació, cal situar el cursor sobre el quadre

gris.

En cas d'abortar, el job finalitzarà en aquest punt. En acceptar (*Proceed*), el job obrirà una petició de canvi a Remedy o correu a SAU (segons si l'aplicació es troba o no a Remedy) a CPD demanant el desplegament d'aquesta a l'entorn de PRE.

Aquest desplegament a l'entorn de PRE serà realitzat per CPD mitjançant un altre job Jenkins. El detall del funcionament d'aquest job de desplegament es troba explicat a l'apartat "5.4. *Jobs desplegament automàtic per a CPD*".

### 5.3.13 Stage SMOKE TEST

En arribar a aquest Stage, el job es tornarà a aturar a l'espera de confirmació manual per continuar una vegada s'hagi rebut confirmació del desplegament a PRE per part de CPD.

ió .D	INT	Smoke Test	PRE	Smoke Test
	1min 5s	66ms	6s	NaNy NaNd
Continuar quan es rebí confirmació de desplegament a PRE.				(paused for 27min 5s)
<input type="button" value="Proceed"/> <input type="button" value="Abort"/>				

Una vegada acceptada la continuació del job, aquesta etapa realitzarà una validació bàsica per detectar que l'aplicació s'ha publicat correctament (com per exemple accedir a una URL de l'aplicació i veure que respon).

### 5.3.14 Stage ACCEPTANCY TEST

Execució dels tests automàtics d'acceptació.

### 5.3.15 Stage EXPLORATORY TEST

Execució dels tests manuals d'acceptació.

### 5.3.16 Stage GENERACIÓ TAG DEFINITIU

Aquesta etapa, com el seu nom indica, genera un TAG, anomenat "versió desplegable", al GitLab. Representa una versió que s'ha compilat, muntat i desplegat correctament, a més d'haver passat tests unitaris, de commit, d'acceptació i exploratoris. És a dir, es tracta d'una versió que ha passat tots els filtres per poder ser desplegada a l'entorn productiu.

La nomenclatura del TAG serà la versió indicada al fitxer /sic/sic.yml que proporciona l'aplicació.

### 5.3.17 Stage PRO

En arribar a aquesta etapa, l'execució del job s'aturarà per demanar confirmació manual per a continuar amb la petició de desplegament a PRO.

	Exploratory Test	Generació Tag DEFINITIU	PRO
	11ms	707ms	2s
Efectuar desplegament a PRO* <input type="button" value="Proceed"/> <input type="button" value="Abort"/>			(paused for 10s)

Per tal que aparegui la finestra modal demanant la confirmació, cal situar el cursor sobre el quadre gris.

En cas d'abortar, el job finalitzarà en aquest punt. En acceptar (*Proceed*), el job obrirà una petició de canvi a Remedy o correu a SAU (segons si l'aplicació es troba o no a Remedy) a CPD demanant el desplegament d'aquesta a l'entorn de PRO.

Aquest desplegament a l'entorn de PRO serà realitzat per CPD mitjançant un altre job Jenkins. El detall del funcionament d'aquest job de desplegament es troba explicat a l'apartat "5. Jobs desplegament automàtic per a CPD".

### 5.3.18 Stage SMOKE TEST

En arribar a aquest Stage, el job es tornarà a aturar a l'espera de confirmació manual per continuar una vegada s'hagi rebut confirmació del desplegament a PRO per part de CPD.

Exploratory Test	Generació Tag DEFINITIU	PRO	Smoke Test
11ms	707ms	6s	NaNy NaNd
Continuar quan es rebí confirmació de desplegament a PRO.* <input type="button" value="Proceed"/> <input type="button" value="Abort"/>			(paused for 39s)


Una vegada acceptada la continuació del job, aquesta etapa realitzarà una validació bàsica per detectar que l'aplicació s'ha publicat correctament (com per exemple accedir a una URL de l'aplicació i veure que respon).

<b>CSCanigó</b>	MANUAL D'USUARI SIC	0192
	SIC Manual Usuari.docx	
	N. versió: 3.0.2	Pàg. 20 / 26

### 5.3.19 Resultats del job i arxivat d'artefactes

Des de que Jenkins fa ús de pipelines, no es guarden els artefactes de les execucions. Si cal recuperar una versió anterior, es recompila el codi des del tag generat.

A efectes de preservació de logs, s'assegura la conservació de les últimes 5 execucions.

	MANUAL D'USUARI SIC	0192
	SIC Manual Usuari.docx	
	N. versió: 3.0.2	Pàg. 21 / 26

## 5.4 Jobs desplegament automàtic per a CPD

Es posarà en disposició dels equips de CPD una sèrie de jobs Jenkins per permetre realitzar el desplegament automàtic de les aplicacions a entorns PRE i PRO.

Aquests jobs només s'hauran d'executar en rebre una petició de desplegament per part dels proveïdors de l'aplicació. En aquesta petició ja s'especificarà el nom del job a executar.

Per dur a terme el desplegament mitjançant el job indicat, caldrà seguir els següents passos:

- Si l'aplicació compta amb BBDD, caldrà realitzar un backup d'aquesta (de l'entorn on es farà el desplegament) prèviament a l'execució del job.
- Accedir al portal de Jenkins (), cercar el job indicat per la petició desplegament i fer click sobre el seu nom
- Dins el job, cal executar-ho mitjançant la opció de menú lateral:



- Abans començar el desplegament, demanarà confirmació sobre la realització del backup de BBDD de l'aplicació a l'entorn de desplegament. Si l'aplicació no compta amb BBDD, caldrà marcar igualment el check.

This build requires parameters:

BACKUP\_BBDD\_REALITZAT ☐

Si l'aplicació disposa de BBDD, cal haver realitzat un backup d'aquesta abans continuar amb el desplegament.

Build

En cas de no marcar el check de Backup i iniciar el desplegament, el job generarà un error i finalitzarà. Altrament, començarà a realitzar el desplegament.

- El desplegament realitzarà les següents accions i en l'ordre indicat:
  - Executarà scripts contra la BBDD si el proveïdor ho ha especificat
  - Desplegarà l'aplicació
  - Enviarà correu notificant finalització del desplegament + el resultat d'aquest a l'usuari que ha executat el job (administrador CPD) + els proveïdors de l'aplicació.

Segons la tecnologia de desenvolupament, característiques de l'aplicació i servidors on s'haurà de desplegar, la tasca de desplegament variarà. Per exemple:


- Per a aplicacions JEE, s'enviarà el contingut estàtic a l'Apache i l'artefacte dinàmic

<b>CSCanigó</b>	MANUAL D'USUARI SIC	0192
	SIC Manual Usuari.docx	
	N. versió: 3.0.2	Pàg. 22 / 26

al servidor d'aplicacions (a excepció de Tomcat, que no cal fer aquest pas perquè el SIC desplega en remot). Posteriorment connectarà amb aquest per deployar l'artefacte/s.

- Per a aplicacions PHP, s'enviarà el contingut a l'Apache.
- Per a aplicacions .NET i .ASP, s'enviarà un paquet amb els binaris + contingut estàtic a l'IIS.

Els desplegaments es realitzaran sobre totes les instàncies de Servidors que disposi l'aplicació a l'entorn.

	MANUAL D'USUARI SIC	0192
	SIC Manual Usuari.docx	
	N. versió: 3.0.2	Pàg. 23 / 26

## 5.5 Execució de scripts de BBDD durant els desplegaments

En el cas que es vulgui executar scripts de BBDD durant els desplegaments automatitzats des de jobs Pipeline, l'usuari haurà de pujar prèviament tant el fitxer de plans com els scripts a un directori independent. A continuació, es mostra un exemple on l'hem col·locat en la carpeta "sql\_scripts":

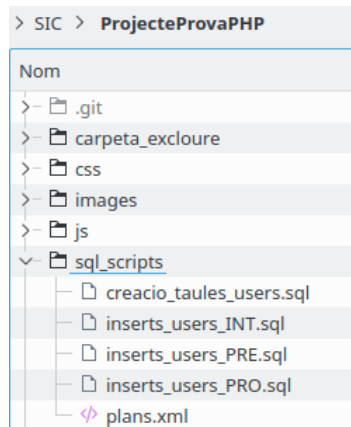



Figura 3.2.6.3 – 1

El format del contingut del fitxer de plans haurà de ser el següent:

```
<llista-scripts>
<script entorn="[ INT | PRE | PRO | FOR ]" failure="[stop/continue]" idBBDD="[identificador de la base de dades]" file="[fitxer1.sql]" />
...
<script entorn="[ INT | PRE | PRO | FOR ]" failure="[stop/continue]" idBBDD="[identificador de la base de dades]" file="[fitxer2.sql]" />
</llista-scripts>
```

Els camps a omplir al fitxer de plans seran els següents:

- **entorn:** Determina per a quin entorn cal utilitzar l'script. Té quatre valors possibles: "INT" (Integració), "PRE" (Preproducció), "PRO" (Producció) i "FOR" (Formació). Es pot fer servir un script per a més d'un entorn. La informació facilitada en aquest atribut s'utilitza tant en els desplegaments automàtics com en els manuals. En els desplegaments automàtics, el servidor Jenkins despleguen els scripts de l'entorn que pertorqui en cada etapa de desplegament. En els desplegaments manuals, empaquetarà els scripts de l'entorn concret per adjuntar-los a la petició de desplegament.
- **failure:** Podrà tenir dos valors, "stop" o "continue". El valor "stop" indicarà que en cas que es produeixi algun error en el processament del script, s'aturarà el procés d'execució de scripts i no s'executarà cap més. El valor "continue" indicarà que tot i que es produeixin errors en el processament del script, el procés d'execució de scripts no s'aturarà.

	MANUAL D'USUARI SIC	0192
	SIC Manual Usuari.docx	
	N. versió: 3.0.2	Pàg. 24 / 26

- **idBBDD:** Identificador de la BBDD on s'executa aquest script. Aquest identificador és un nom arbitrari decidit pel proveïdor d'aplicacions. Pot ser un nom indicatiu de la base de dades a la qual s'ha de connectar l'aplicació. Pot haver-hi tants identificadors com bases de dades utilitzi l'aplicació. Exemples: (sqlserver\_dwh\_int, sqlserver\_trans\_pro, oracle\_corp1\_pre, oracle\_ens\_int, etc.).
- **file:** A part d'indicar el número del script, el departament, l'aplicació i una breu descripció del que fa el script (per exemple: insert, update, delete, create, etc), caldrà informar el fitxer com a ".sql" en el cas que es tracti un script SQL i com a ".pl" en el cas que es tracti d'un fitxer amb scripts PL/SQL. Serà molt important indicar la correcta extensió del fitxer segons el contingut, ja que sinó el processament serà erroni.

- 

Un exemple del format seria el següent:

- **Fitxer de plans:** *CTTI\_test\_plans\_INT.xml*
- **Contingut:**

```
<llista-scripts>
<script entorn="INT" failure="stop" idBBDD="oracle_int" file="creacio_taulers_users.sql"/>
<script entorn="INT" failure="continue" idBBDD="oracle_int" file="inserts_users_INT.sql"/>
<script entorn="PRE" failure="stop" idBBDD="oracle_pre" file="creacio_taulers_users.sql"/>
<script entorn="PRE" failure="continue" idBBDD="oracle_pre" file="inserts_users_PRE.sql"/>
<script entorn="PRO" failure="stop" idBBDD="oracle_pro" file="creacio_taulers_users.sql"/>
<script entorn="PRO" failure="stop" idBBDD="oracle_pro" file="inserts_users_PRO.sql"/>
</llista-scripts>
```

Pel que fa al contingut dels scripts, caldrà seguir una lògica segons quin tipus es faci servir (SQL o PL/SQL) i el nom del fitxer de scripts.

En el cas dels fitxers amb scripts PL/SQL serà imprescindible per la seva correcta execució, que es finalitzi el script correctament informant al final del script el següent:


```
/
EXIT;
```

Abans d'executar el script es comprovarà això i, en cas que no estigui indicat aquest final de fitxer, aquest no és processarà.

Un exemple de com quedaria el script PL/SQL seria el següent:

```
DECLARE
variable1 CHAR(50) := 'Test1';
variable2 CHAR(50) := Test2;
```



	MANUAL D'USUARI SIC	0192
	SIC Manual Usuari.docx	
	N. versió: 3.0.2	Pàg. 25 / 26


```

variable3 CHAR(50) := 'Valor1';
BEGIN
UPDATE taula_test
SET valor = variable3
WHERE prova1 = variable1
AND prova2 = variable2;
COMMIT;
END;
/
EXIT;

```

Durant la integració de l'aplicació al SIC, es sol·licitarà al proveïdor d'aplicacions les dades necessàries corresponents a cada identificador de Base de Dades (cadena de connexió, servidor, port, etc.).

**Important:** El job Pipeline es llançarà de manera automàtica al detectar una actualització de codi al repositori Gitlab de l'aplicació. Si existeix la carpeta d'scripts + el fitxer de plans al codi, es durà a terme l'execució dels scripts BBDD definits en aquest fitxer. Per tant, **és important tenir en compte abans cada pujada de codi el mantenir, actualitzar o treure la carpeta amb el fitxer de plans i scripts de BBDD.**

	MANUAL D'USUARI SIC	0192
	SIC Manual Usuari.docx	
	N. versió: 3.0.2	Pàg. 26 / 26

## 5.6 Instal·lació de llibreries JEE, Microsoft i mòduls npm i bower


### 5.6.1 Introducció

Aquest Job compila, construeix e instal·la una llibreria en el repositori local del SIC amb l'objectiu que sigui utilitzada per altres aplicacions en el SIC.

El nom del job aconsegueix la següent sintaxi: `[codiDiàleg]-[nomLlibreria o nomMòdul]`.

Els projectes de llibreries/mòduls han de ser projectes independents, ja que són susceptibles de ser versionats independentment de l'aplicació que els utilitza. Per tant, s'hauran de crear en un repositori exclusiu per a cada llibreria/mòdul.

### 5.6.2 Execució del job

El procediment per l'execució d'aquest build és anàleg a la resta de Jobs. Hi ha dues formes de fer-ho, des de la llista de builds (figura 3.2.3 – 1) amb el botó  o bé amb el link “Build now” des de la pàgina principal del build (figura 3.2.4 – 1).

En accedir al formulari d'execució es demanarà la versió de la llibreria a compilar. Aquest valor a de coincidir amb el nom del tag on estigui el codi de la llibreria que es vulgui compilar.

L'execució satisfactòria d'aquest job farà que es compili i s'instal·li la llibreria en el repositori local del Hudson de forma que sigui utilitzable per altres aplicacions que estiguin al SIC.

## 6 Format de l'arxiu sic.yml

Actualment, aquest arxiu només inclou la versió de l'aplicació conforme a la nomenclatura de versions de CTTI (`<versióMajor>.<versióMenor>.<versióFix>`). Per tant, un exemple d'arxiu vàlid és el següent:

```
version: 1.0.1
```