Setup de ConfigMap y Microservicios nuevos en Gluon

Esta guía pretende incluir los pasos a dar con nuevos desarrollos de Gluon a la hora de la configuración inicial del CM (ConfigMap), Micro y Openshift entre otros. Puede contener errores o puede que le falte información porque se ha hecho recopilando información de diferentes fuentes a lo largo del proceso de configuración.

Se va a incluir información de la documentación (de la cual se agregarán links) e información sacada de fuera de la documentación a lo largo del proceso (equipos de Teams, conversaciones, etc.).

# Consideraciones Iniciales

Deberían estar creados los repositorios que contienen:

El Micro: <https://github.com/santander-group-spain-gln/san-sapawa-sanpaywallet>

El CM: https://github.com/santander-group-spain-gln/san-sapawa-cmsanpaywallet

El OAM: https://github.com/santander-group-spain-gln/san-sapawa-oamsapawa

Aquí vamos a ver lo que cuelga del Developer, CM y Micro. Para más info sobre el OAM, hay documentación (por ejemplo [aquí](https://gluon.gs.corp/community/docs/latest/components/software/api/framework/infrastructure/oam-configuration/)).

# Configuración del CM

En este apartado veremos lo que hay que cambiar en el componente. Tendremos que tocar sobre los directorios **.gluon** y **config**.

Sobre esto podéis encontrar más información (inglés) en [este link.](https://gluon.gs.corp/community/docs/latest/components/configuration/kubernetes/configmaps-rm/#configmap-to-file)

## Directorio .gluon

Aquí encontraremos un directorio **CD** y un directorio **CI**.En **CD** encontramos **un directorio para cada entorno** (cert, pre, pro) y un archivo **values.yaml**. Este archivo contendrá las propiedades genéricas del CM.

Cada uno de los directorios de entorno contendrá dos archivos:

**cd.yml**

**values-[entorno].yaml**

De estos dos, tendremos que hacer cambios en el archivo **cd.yml** de cada entorno, explicados a continuación.

### Directorio CD

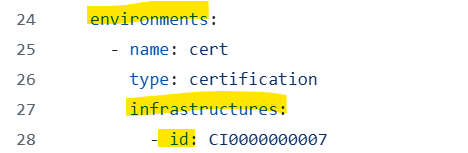
Este directorio contiene la configuración genérica y la configuración por entorno.

#### Archivo cd.yml

Sobre este archivo tendremos que aplicar cambios en la propiedad **ci\_id**. Normalmente llegará con un placeholder (OAM-CI-ID). El valor de esta propiedad se encuentra en el OAM.

En el OAM, tendremos que abrir el archivo **oam.application-definition.yml** y buscar:

environments -> infrastructures -> id



En este caso es para cert, pero deberían estar todos los entornos en el archivo.

#### Resto de archivos en directorio CD

Sobre el archivo values-[entorno].yaml de cada entorno no aplicamos cambios.

Sobre el archivo values.yaml genérico no aplicamos cambios.

Ambos deben tener de inicio una forma parecida a esta:

Values-[entorno].yaml

Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Values.yaml

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

#### NOTA IMPORTANTE SOBRE DIRECTORIO **PRO**

El directorio PRO de la carpeta .gluon **hay que dividirlo en dos**: **pro-b** y **pro-g.**

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Además, en los archivos **cd.yml** de los dos nuevos directorioshay que añadir la propiedad **component\_name** además de la configuración anterior (ci\_id). También hay que adaptar la ruta de uno de los valores de **configuration\_files** para que la ruta apunte bien al values-pro.yaml correspondiente.

Quedaría algo así para el cd.yml de pro-b:

Texto

El contenido generado por IA puede ser incorrecto.

### Directorio CI

En este directorio encontraremos un **properties.env** que estará vacío y dejaremos así.

## Directorio config

Aquí encontraremos **una carpeta por cada entorno**.

Cada una de estas carpetas tendrá un **application-[entorno].properties**. Igual que en .gluon -> cd, aquí tenemos que **dividir la carpeta PRO en dos** (pro-b y pro-g), cada una contendrá su application-pro.properties, quedando algo así:

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

### Archivo application-[entorno].properties

Estos archivos contendrán las properties necesarias para el funcionamiento del micro unido al CM en cada entorno.

Se recomienda coger el **properties.local** que deberíamos tener en nuestro micro y **adaptarlo a cada uno de los entornos**.

Tanto para este como para cualquier otro properties, **las credenciales siempre en secret**.

## Despliegue del CM

Para desplegar el CM iremos a **actions**, dentro del repo, e iremos al **workflow** que queramos ejecutar (**Deploy** en este caso). Cuando se abra la ventana de Deploy, clicaremos en **Run Workflow** y rellenaremos el desplegable:

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

En cuanto a las opciones del desplegable, las rellenaremos de la siguiente manera. Los campos en verde son obligatorios.

Version -> No aplica en los CM

**Environment to deploy** -> cert/pre/pro

**Environment type to deploy** -> certification, preproduction, production

SNOW task number -> No aplica

## Notas extra sobre CM

Esta información ha sido obtenida de medios ajenos a la documentación, como el [Equipo de Cards en Teams](https://teams.microsoft.com/l/team/19%3AFPZNlQ7C8rGA_6zFe5fnItNDBF5IvCwvIS4WMmkKLGA1%40thread.tacv2/conversations?groupId=b0620f36-3f87-40d2-99f1-9658d4fdc023&tenantId=35595a02-4d6d-44ac-99e1-f9ab4cd872db), donde se publican cambios o cosas a adaptar tanto en Micros como en CMs referentes a Gluon, entre otras cosas.

**Cada vez que se crea un componente Gluon se debe añadir en GitHub los *secrets* necesarios para el despliegue**. Para ello debéis abrir Jira a DevOps incluyendo el link del componente a configurar.

Jira de Devops: <https://sanes.atlassian.net/browse/HUBCUSPRDJ>

Ejemplo de Ticket:



[Link al mensaje en el Equipo de Cards](https://teams.microsoft.com/l/message/19:FPZNlQ7C8rGA_6zFe5fnItNDBF5IvCwvIS4WMmkKLGA1@thread.tacv2/1747733471365?tenantId=35595a02-4d6d-44ac-99e1-f9ab4cd872db&groupId=b0620f36-3f87-40d2-99f1-9658d4fdc023&parentMessageId=1747733471365&teamName=Cards&channelName=General&createdTime=1747733471365)

Para securizar los logs de Gravity en componentes **Gluon (Micro)** se debe añadir lo siguiente en el fichero **values.yaml** (hay que añadir también los **SECRETS**):

extraEnvVarsSecret: gravity-kafka

extraEnvVarsCM: cm-gravity-kafka

Y en el fichero **\*.properties**del config-map (CM) asociado al componente hay que añadir lo siguiente:

spring.cloud.config.allow-override = true

spring.cloud.config.override-system-properties = false

[Link al mensaje en el Equipo de Cards](https://teams.microsoft.com/l/message/19:FPZNlQ7C8rGA_6zFe5fnItNDBF5IvCwvIS4WMmkKLGA1@thread.tacv2/1740054862582?tenantId=35595a02-4d6d-44ac-99e1-f9ab4cd872db&groupId=b0620f36-3f87-40d2-99f1-9658d4fdc023&parentMessageId=1740054862582&teamName=Cards&channelName=General&createdTime=1740054862582)

## Posibles errores al lanzar un deploy

Si encontramos este error:

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

Se puede deber a problemas con los namespaces en Maia. Hay que abrir ticket a Devops para que lo solucionen.

# Configuración del Microservicio

En este apartado veremos qué hay que crear/adaptar en el microservicio de Gluon para completar el setup. Tocaremos sobretodo la configuración del directorio **.gluon**.

Para lo que vamos a ver a continuación, nos podemos apoyar en parte en [este link.](https://sanes.atlassian.net/wiki/spaces/ESPHIPOTEC/pages/26121339784/Ficheros+de+Configuraci+n.)

## Directorio .gluon

Este directorio es parecido al del CM. Encontraremos los directorios **CD** y **CI**, conteniendo el primero las **configuraciones por entorno** y el segundo el **properties.env**.

### Directorio CD

Este directorio **es igual** que el homónimo en el CM. La única diferencia es que aquí también tenemos un **values-native.yaml**, que sirve para la **configuración de los recursos de memoria** del micro. Dentro de cada carpeta de un entorno encontramos los archivos **cd.yml** y **values-[entorno].yaml.**

#### Archivo cd.yml

Como en el CM, hay que colocar bien el **ci\_id** sacándolo del **OAM.**

#### Archivo values-native.yaml

Este archivo como se comenta con anterioridad sirve para la configuración de los **recursos de memoria del micro**. A priori no hay que tocar nada.

#### Archivo values.yaml

En este archivo debemos añadir ciertos cambios. El primero es añadir las siguientes properties:  
name: ${APPLICATION\_NAME}

nameOverride: ${APPLICATION\_NAME}

fullnameOverride: ${APPLICATION\_NAME}  
  
Esto sirve para enlazar el ConfigMap con el Microservicio.

Más información [aquí.](https://sanes.atlassian.net/wiki/spaces/ESDEMISION/pages/26372014457/Cambios+en+CM+2.0+para+OCU+PRO#Anexo%3A-Enlazar-config-map-a-microservicio)

Por otro lado, si tenemos que añadir **datos de variables de entorno** para que se vean reflejadas en Openshift, se añadirán bajo la property ***extraEnvVars.***

Dejo un ejemplo a continuación donde añado valores para las properties GRAVITYCM-USERNAME y GRAVITYCM-PASSWORD:

extraEnvVars:

- name: GRAVITYCM-USERNAME

valueFrom:

secretKeyRef:

name: pwd-gravity-config-manager

key: "username"

optional: false

- name: GRAVITYCM-PASSWORD

valueFrom:

secretKeyRef:

name: pwd-gravity-config-manager

key: "password"

optional: false

Y aquí se ve cómo se usan en el **application-[entorno].properties** correspondiente del CM:  
Texto

El contenido generado por IA puede ser incorrecto.

También habrá que revisar en Openshift si están creados los **volúmenes de krb5 y del keytab**. Si no lo están, habrá que añadir lo siguiente en las properties **extraVolumes** y **extraVolumeMounts**.

extraVolumes:  
 - name: secret-sapawa-keytab  
 secret:  
 secretName: secret-sapawa-keytab  
 defaultMode: 420  
 - name: cm-sapawa-krb5  
 configMap:  
 name: cm-sapawa-krb5  
 defaultMode: 420

extraVolumeMounts:  
 - name: secret-sapawa-keytab  
 mountPath: /etc/keytab-volume  
 - name: cm-sapawa-krb5  
 mountPath: /etc/krb

Cuando despleguemos el micro, podemos ir a **Openshift** y comprobar que haya leído bien las properties y haya generado los volumes en los **deployment details**.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

NOTA: Sobre **la creación de los secrets** en Openshift y de las **variables de entorno** habrá otro apartado al final.

En [este mensaje del Equipo de Cards](https://teams.microsoft.com/l/message/19:FPZNlQ7C8rGA_6zFe5fnItNDBF5IvCwvIS4WMmkKLGA1@thread.tacv2/1748357176659?tenantId=35595a02-4d6d-44ac-99e1-f9ab4cd872db&groupId=b0620f36-3f87-40d2-99f1-9658d4fdc023&parentMessageId=1729687734346&teamName=Cards&channelName=General&createdTime=1748357176659) se explica que también hay que añadir el siguiente cambio por problemas de memoria en el **values.yaml de todos los entornos** (pero sobretodo DEV).

resources:  
 limits:  
    memory:1500M  
    cpu: 1000m  
 requests:  
    memory: 975M  
    cpu: 100m

En el directorio de **PRO**, tal y como se indica en [este mensaje del Equipo de Cards](https://teams.microsoft.com/l/message/19:FPZNlQ7C8rGA_6zFe5fnItNDBF5IvCwvIS4WMmkKLGA1@thread.tacv2/1750317619681?tenantId=35595a02-4d6d-44ac-99e1-f9ab4cd872db&groupId=b0620f36-3f87-40d2-99f1-9658d4fdc023&parentMessageId=1750317619681&teamName=Cards&channelName=General&createdTime=1750317619681), metemos en el **values-pro.yaml** lo siguiente:

replicaCount: 0

logging\_level\_root: ERROR (esta cuelga de la property “**darwin**”)

Quedará así:  
Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

Dentro del mensaje citado arriba hay más información de por qué se hace este cambio.

#### Otros añadidos o fixes

Si nuestro microservicio usa **GRAVITY**, hay que añadir [este fix](https://teams.microsoft.com/l/message/19:FPZNlQ7C8rGA_6zFe5fnItNDBF5IvCwvIS4WMmkKLGA1@thread.tacv2/1738672748246?tenantId=35595a02-4d6d-44ac-99e1-f9ab4cd872db&groupId=b0620f36-3f87-40d2-99f1-9658d4fdc023&parentMessageId=1738672748246&teamName=Cards&channelName=General&createdTime=1738672748246) al **application.yml**.

Cambiar:

app.configmanager.artifactName: ${PROJECT\_NAME}-${env.console.gravity.environment}\_${spring.application.name}

por

app.configmanager.artifactName: ${PROJECT\_NAME}\_[NOMBRE DEL MICRO]

Por ejemplo, para el micro **san-sapawa-sanpaywallet** quedaría así:

Texto

El contenido generado por IA puede ser incorrecto.

### Directorio CI

Este archivo debería venir pregenerado con los datos siguientes. Si es así, no habría que tocar nada:

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

# Creación de secrets y variables de entorno en Openshift (Dev)

En este apartado se va a explicar cómo crear secrets y variables de entorno en Openshift (los desarrolladores solo pueden hacerlo en entorno Dev/Cert).

Para secrets, se van a explicar dos ejemplos:

Un secret de credenciales (usuario y contraseña)

Un secret de objeto

## Secrets

Los secrets nos sirven, entre otras cosas, para **ofuscar detalles sensibles de nuestros properties de entorno**, como usuario/contraseña de BBDD, la URL de conexión, etc.

### Secret de credenciales

Un secret de credenciales nos permite ofuscar el login de acceso a cualquier servicio que tengamos en nuestro **properties-[entorno].yaml** **del CM**.

En este ejemplo usaremos las credenciales de DB2 que se encuentran en los properties de cada entorno:  
Imagen que contiene Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.

Tal y como vemos en la siguiente imagen, tenemos que ir a **nuestro proyecto** (en el ejemplo sanes-sapawa-sanpaywallet) en Openshift, luego a **Secrets** y por último a **Create**.

Tabla

El contenido generado por IA puede ser incorrecto.

Nos saldrá un desplegable en el que elegiremos **From YAML.** En la ventana con las properties, rellenamos lo siguiente:

name: el nombre del secret, por ejemplo **pwd-db2**

type: aquí el tipo de secreto, para credenciales se puede usar **kubernetes.io/basic-auth** pero también se crean algunos con type **Opaque**.

username: el valor del **usuario** de las credenciales

password: el valor de la **contraseña** de las credenciales

Para finalizar, clicamos abajo en **Create.**

Debería quedarnos algo como esto:

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Podemos comprobar en **reveal values** (en la ventana del secret, abajo a la derecha) que se haya creado bien.

### Secrets de objeto

Si, como vimos con anterioridad en [este documento](#_Archivo_values.yaml), hay que crear un **volume para el keytab**, también habrá que crear un **secret para guardarlo** (el keytab al fin y al cabo es un archivo) y que se pueda acceder a él.

Siguiendo los pasos del secret de credenciales, esta vez en **Create** iremos a **Key/Value Secret**.

En la ventana que se abre, rellenaremos con lo siguiente:

**Secret name:** el **nombre del secret**, por ejemplo **secret-sapawa-keytab**

**Key:** **la clave del secret**, por ejemplo en este caso el nombre del keytab

**Value:** aquí podemos usar **Browse…** para buscar el archivo (keytab), o **arrastrarlo directamente** para subirlo

Al final nos debería quedar algo así:

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

Podemos comprobar que ha cogido bien el valor del keytab yendo a **la pestaña YAML** y comprobando la property clave/valor que se ha creado.

## Variables de entorno

Las variables de entorno son las que recogen nuestros secrets en cada entorno y **leen sus valores**.

Hay diferentes maneras de crear variables de entorno, aquí se va a mostrar cómo crearlas **desde ConfigMap o Secret**.

En Openshift, vamos a nuestro Microservicio desde **Deployments** en nuestro Proyecto:

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

En este caso, san-sapawa-sanpaywallet:

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

Una vez en nuestro deployment del microservicio, vamos a **Environment** y encontraremos todas las variables de entorno ya creadas o disponibles.

Esto se rellena **con las variables de entorno que se leen en el values.yaml del microservicio**, por ejemplo las que vimos en el apartado de [Archivo values.yaml](#_Archivo_values.yaml).

Pero además, tenemos que darles un valor, es decir, **unirlas a los secret**.

Clicamos en la siguiente opción:

Imagen que contiene Sitio web

El contenido generado por IA puede ser incorrecto.

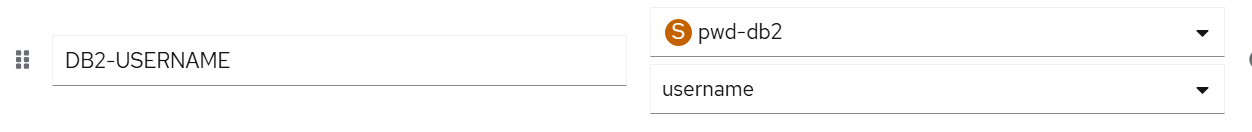
En los campos que aparecen, rellenamos con lo siguiente:

**Name:** La variable de entorno tal y como está en el properties, por ejemplo, **DB2-USERNAME**

**Select a resource:** Aquí tenemos que elegir entre CONFIGMAPS o SECRETS, **en nuestro caso queremos un secret** así que buscaremos el que hemos creado antes, **pwd-db2**

**Select a key:** Aquí temenos que elegir **uno de los campos del secret**, en este caso el **username**

Al terminar, debería quedar algo así:



# Bibliografía

A continuación, se añaden links a todas las partes de la documentación que se han consultado en mayor o menor medida. Algunos de estos links están también distribuidos por el documento en ciertos lugares para ayudar con algo concreto.

## Documentación de Gluon (Inglés)

[Docu sobre Deployments/Workflow](https://gluon.gs.corp/community/docs/latest/application/ci-cd/cd/cd-rm/cd-workflow/)

[Docu sobre directorio CD del Micro](https://gluon.gs.corp/community/docs/latest/application/ci-cd/cd/cd-rm/cd-workflow/cd-envs-configuration/)

[Docu sobre directorio CI del Micro](https://gluon.gs.corp/community/docs/latest/application/ci-cd/cd/cd-rm/cd-workflow/ci-envs-configuration/)

[Docu sobre config de Helm](https://gluon.gs.corp/community/docs/latest/components/software/backend/java/darwin/framework/darwin-helm/)

[Docu general sobre configuración del Micro](https://gluon.gs.corp/community/docs/latest/components/software/backend/java/darwin/darwin-maven-kubernetes-rm/)

[Docu general sobre configuración de los ConfigMaps](https://gluon.gs.corp/community/docs/latest/components/configuration/kubernetes/configmaps-rm/)

[Docu sobre secrets en GLUON](https://gluon.gs.corp/community/docs/latest/components/configuration/kubernetes/application-secrets/#introduction)

## Documentación de la Wiki (Español)

[Docu sobre ficheros de configuración](https://sanes.atlassian.net/wiki/spaces/ESPHIPOTEC/pages/26121339784/Ficheros+de+Configuraci+n.)

[Docu sobre creación de secrets (Openshift, Github...)](https://sanes.atlassian.net/wiki/spaces/ESPVALORES/pages/25216254993/Creaci+n+de+secrets)

[Docu sobre creación de secrets en GLUON](https://sanes.atlassian.net/wiki/spaces/ESPGDPROD/pages/26083000649/Kubernetes+Secret+en+Gluon)

[Cambios sobre CM 2.0 en OCU y PRO](https://sanes.atlassian.net/wiki/spaces/ESDEMISION/pages/26372014457/Cambios+en+CM+2.0+para+OCU+PRO)

[Configuración en Micros creados con modelo OAM (2025)](https://sanes.atlassian.net/wiki/spaces/FINANFACT/pages/26170949647/Micros+creados+bajo+modelo+OAM+a+partir+de+2025)

## Documentación genérica

[Tipos de secretos en Kubernetes/Openshift](https://kubernetes.io/docs/concepts/configuration/secret/#secret-types)