Actualización del Sistema Osmosys

# Actualización del Sistema Osmosys

Antes de iniciar la actualización, es fundamental realizar respaldos completos de los componentes actuales del sistema, estos respaldos servirán en caso de tener problemas durante el proceso de instalación o posterior a las pruebas funcionales.

Considerar que se debe respaldar por las iniciales de la operación cuando se encuentre lso caracteres XXX en este manual, un ejemplo de las operaciones actuales existentes es:

* Ecuador: ecu
* El Salvador: slv
* Costa Rica: cri
* México: mex
* Perú: per
* Venezuela: ven
* Buro: rba
* Test: test

Pasos para el respaldo:

**1. Respaldo de la base de datos.**

EL resplado se lo puede hacer directamente desde PGAdmin.

**Copiar la carpeta de Frontend de la version antigua a un directorio seguro.**

* Comprimir con el comando zip la carpeta dentro de la carpeta del frontend que se ubica dentro de apache:  
  zip -r xxx-old.zip /var/www/html/xxx
* Respaldar el archivo comprimido

**Respaldar el artefacto WAR de la version antigua.**

* Se debe copiar el archivo war de la versión antigua junto con el respaldo del archivo comprimido del frontend.
* La ubicación del archivo war es
  + /opt/wildfly/standalone/deployments/

**2. Actualización de artefactos y carpeta de distribución**

Una vez realizados los respaldos, procederemos con la actualización de la versión de Osmosys.

**Ejecución de los scripts para actualizar la estructura de datos y otros parámetros importantes.**

Es importante ejecutar los siguientes pasos en el orden especificado

* **Actualización de estructura**
  + Ejecutamos el **script install\_structure.sql.**
* **Migración de datos y actualización de parámetros generales.**
  + Ejecutamos el script **migración\_xxx.sql**
  + Se debe ejecutar el script considerando la operación
* **Creación de nuevas dimensiones para el cubo de información**
  + Ejecutamos el script **cube\_creation\_views.sql**

**3. Generación de Frontend**

Con la estructura de datos actualizada, es momento de generar los artefactos y las carpetas de distribución para la versión nueva.

Pasos para la creación de artefactos:

* Crear todos los artefactos (backend) y carpetas de distribución (frontend) necesarios para la versión nueva a partir de la rama main.

Considerar respaldar cambios del código antes de hacer un pull al branch main con el siguiente comando

* + git pull origin main
* Configuración de variables de frontend
  + Se debe configurar las variables necesarias dependiendo de la operación y del ambiente a desplegar, el archivo de configuración es **environment.prod\_xxx.ts**, ubicado en el path: osmosys/frontend/src/environments, las configuraciones se realizan según las siguientes caracteristicas:
    - production.- true o false dependiendo del ambiente, en el caso de ser false el pie de las pantallas (footer) si pinta de color amarillo para no confundir con un ambiente de producción.
    - hrm.- false
    - locale.- es
    - base\_url: ‘https://backend.osmosys.unhcr.org:8443/osmosys\_xxx/api'

dirección del backend

* + - app\_code: 'OSMOSYS\_XXX', remplazar XXX por el acronimo de la operación
    - versionCheckURL: ‘https://osmosys.unhcr.org/rba/version.json’
    - debe ser la dirección de acceso por navegador a Osmosys más /version.json
    - loginPhoto : 'unhcr\_photo\_ec.png'
    - flagLoginFile : 'flag\_test.png',
    - flagToolbarFile : 'flag\_test.png',
    - operationName : 'XXX'
  + Un ejemplo válido para la operación RBA sería el siguiente:

production: true,

hmr: false,

locale: 'es',

base\_url: 'https://backend.osmosys.unhcr.org:8443/osmosys\_rba/api',

app\_code: 'OSMOSYS\_RBA',

versionCheckURL : 'https://osmosys.unhcr.org/rba/version.json',

loginPhoto : 'unhcr\_photo\_ec.png',

flagLoginFile : 'flag\_test.png',

flagToolbarFile : 'flag\_test.png',

operationName : 'RBA'

* Crear distribución para frontend
  + Revisamos la configuración dentro de **angular.json,** cada operación tiene un segmento de configuración, se debe revisar especialmente los siguientes items
    - baseHref .- en esta configuración se debe especificar la ruta de acceso (contexto) que deberán ingresar en el navegador los usuarios de Osmosys. Como ejemplo para esta configuración, para RBA sería la siguiente: **"baseHref": "/rba/",**con esta configuración produce que el acceso a Osmosys sea por la siguiente URL: https://osmosys.unhcr.org**/rba/**
  + Ejecutamos el siguiente comando (remplazar xxx):
    - ng build --configuration=production\_xxx
  + Creamos archivo de version con el comando:
    - node ./build/post-build-operation.js xxx
  + Esto genera la carpeta para la distribución en el path /dist/XXX

Debemos comprimir esta carpeta y publicarla dentro del servidor apache en el servidor a desplegar.

* En el caso que se quiera obtener las carpetas de distribución para todas las operaciones, se puede usar los siguientes comandos,
  + npm run build:all
  + ./build-all-operation.sh

Esto crea un archivo dist.zip mismos que ahorra tiempo obteniendo un comprimido final con todas las operaciones que se deberán descomprimir directamente en el servidor Apache.

En el caso de nuevas operaciones se debe:

* + Crear un nuevo segmento de configuraciones dentro de **angular.js**
  + Agregar la compilación en el archivo **package.json** en el script build:all de acuerdo a la nueva operación.
  + Agregar script dentro del archivo **build-all-operation.sh**

**4. Crear el artefacto WAR para la nueva versión.**

* + Verificar que estemos en el ramal correcto
  + Dentro del archivo **pom.xml** tenemos un segmento por cada operación dentro de este debemos considerar los siguientes apartados:
    - datasource.name: nombre de la base de datos
    - context-root: contexto de cada operación, un ejemplo es osmosys\_xxx
    - hibernate.show\_sql, true si queremos que se presenten las consultas sql en el log de wildfly.
    - warName: nombre del war a generar, un ejemplo sería osmosys\_xxx
    - output.dir: path del Directorio donde se generará el archivo war, un ejemplo para este caso sería ${project.build.directory}/xxx
    - Un ejemplo completo para este segmento sería:

<profile>

<id>xxx</id>

<properties>

<datasource.name>osmosys\_xxx</datasource.name>

<context-root>osmosys\_xxx</context-root>

<hibernate.show\_sql>false</hibernate.show\_sql>

<warName>osmosys\_xxx</warName>

<output.dir>${project.build.directory}/xxx</output.dir>

</properties>

</profile>

* + Compilación de archivo war.
    - Luego de estas configuraciones se debe compilar el archivo war con los siguienes comandos:
      * mvn clean
      * mvn package xxx

Esto generará el archivo war dentro de la carpeta target/ según el nombre especificado con osmosys\_xxx.

* En el caso de necesitar generar los archivos war para todas las operaciones se puede ejecutar el comando ./package\_all.sh mismo que genera un archivo comprimido que contiene los archivos war para todas las operaciones y que deberán ser desplegados en el servidor wildfly.

En el caso de nuevas operaciones se debe:

* + Agregar el acronimo xxx dentro de la línea

profiles=("cri" "ecu" "mex" "per" "slv" "test" "ven" "rba")

en el archive ./package\_all.sh

* + Se debe agregar un nuevo segmento dentro de **pom.xml.**

**5. Configuración de archivo standalone.xml**

* En el archivo standalone.xml se debe agregar las conexiones a las bases de datos de cada operación según las siguientes consideraciones:
  + jndi-name: nombre del jdni configurado
  + connection-url: la cadena de conexiónque apunta a la base de datos un ejemplo sería: jdbc:postgresql://localhost:5432/osmosys
  + user-name: username de la base de datos Postgres
  + password: password de la base de datos.

Cada operación mantiene su propia base de datos, por lo cual cada operación debe tener un segmento de datasource, un ejemplo completo sería el siguiente:

<datasource jta="true" jndi-name="java:jboss/datasources/osmosys\_xxx" pool-name="osmosys\_xxx" enabled="true" use-ccm="true" statistics-enabled="true">

<connection-url>jdbc:postgresql://localhost:5432/osmosys\_xxx</connection-url>

<driver>postgresql-driver</driver>

<pool>

<max-pool-size>100</max-pool-size>

</pool>

<security>

<user-name>username</user-name>

<password>password</password>

</security>

<validation>

<valid-connection-checker class-name="org.jboss.jca.adapters.jdbc.extensions.postgres.PostgreSQLValidConnectionChecker"/>

<background-validation>true</background-validation>

<exception-sorter class-name="org.jboss.jca.adapters.jdbc.extensions.postgres.PostgreSQLExceptionSorter"/>

</validation>

</datasource>

**6. Reiniciar el servidor:**

Como paso final es necesario reiniciar los servidores.

* Apache : sudo systemctl restart apache2
* Wildfly: sudo systemctl restart wildfly

**7. Reverso:**

En caso de que se presenten problemas durante la actualización, es posible revertir los cambios a la versión anterior restaurando los respaldos realizados:

* Restaurar el respaldo de la base de datos.
* Restaurar la carpeta de distribución de la versión anterior dentro del servidor Apache.
* Colocar el artefacto WAR respaldado de la versión anterior en el lugar correspondiente.
* Reinicio de servidores Apache y Wildfly

**8. Consideraciones:**

El sistema incluye la presentación de dashboards en Power BI, es necesario realizar los siguientes ajustes:

1. Actualizar los datasets.

2. Cambiar la ubicación de los datos de los responsables de proyecto (puntos focales).