# INSTRUCTIVO

# PARA USO DEL GIT

## Anexo

Gobierno de la Ciudad Autónoma de Buenos Aires

**OBJETIVO:**

El objetivo del presente documento es poder unificar las entregas en una sola herramienta de control de versiones o SVC (del inglés *System Version Control*)), a través de un proceso y estructura definidos que permitan a las diferentes áreas de la ASI realizar las actividades e intercambios con los proveedores. Es de carácter obligatorio realizar este proceso para llevar a cabo las actividades de control de cambios tanto para aplicaciones nuevas como entregas de nuevas versiones deaplicaciones existentes.

Como aclaración informarmos que éstas entregas no reemplazan la entrega formal por Mesa de Entradas en contrataciones licitadas, sino que es un medio adicional obligatorio para facilitar las tareas operativas.

**ALCANCE:**

El alcance del presente documento involucra a todos los Organismos que interectúan con la ASI, así comolos proveedores del GCABA.

**PROCEDIMIENTO**

El sistema de control de versiones seleccionado por la ASI a utilizar es GIT el cuál pertenece al tipode arquitectura de repositorios de información distribuidos.
Todo el código fuente necesario para el proyecto así como también para el esquema, sus migraciones y todo el material generado para el proyecto deberá estar presente en un repositorio creado a tales efectos.

### Estructura de proyecto:

Cada aplicación debe contar con su carpeta correspondiente y la estructura definida.

NOMBRE PROYECTO

- build/
- database/
    - database.sql
    - migrations
    - version-name-(orden).sql
    - ...
- source/
- documents/
- Instructivos-Documentos ASI/
- scripts/ (opcional) ej: Scripts para rollback, Scripts TestServices.sh
- tests/
- patches/
- .gitignore
- .gitmodules (opcional)
- Makefile (opcional)
- README.md
- CHANGELOG.md
- requirements.txt

En la Carpeta ***Build/*** se deben incluir todos los binarios compilados en caso de tratarse de un

lenguaje compilado.
En la carpeta ***Documents/*** incluir en cada entrega la documentación relacionada a la versión o el proyecto que se encuentra definida como obligatoria en el Estándar de Desarrollo (Ej.: documento de arquitectura, casos de prueba, manuales, etc.). Para realizar la primera instalación es imprescindible contar con el Manual de Instalación completo y el Documento de Arquitectura(Solicitar los template a la ASI como referencia)
En la carpeta ***source/*** debe estar contenido todo el código fuente de la aplicación y el archivo de configuración de dependencias, el cuál debe listar las mismas especificando el número de versión exacta para cada una de ellas.

En caso de necesitar aplicar algun tipo de parche al framework utilizado los mismos deberan alojarse dentro de la carpeta patches de la raíz del repositorio.

En el directorio raíz del repositorio deberá encontrarse un archivo ***README.md*** el cual debe contener la información necesaria para realizar la implementación del aplicativo.
Luego, para cada nueva versión que se entrega es obligatorio sumar al archivo ***README.md*** las instrucciones para llevar a cabo la instalación, indicando la fecha en la que se produjo dicha actualización y los cambios que se realizaron en el archivo ***CHANGELOG.md***, incluyendo en el mismo lo siguiente:

- Nro de tickets de bugs resueltos (nro asociado en la Herramienta de seguimiento y control de cambios definidas en el proyecto)
- Funcionalidades incluídas (si corresponde),
- Sprint correspondiente (versión preliminar).

Cada versión deberá estar etiquetada con correspondiente nombre con el formato **"vx.x.x".**

Cuando el Referente del proyecto de la ASI tiene la aprobación(del Usuario y QA si interviene en le proceso) de la versión pasar de ambiente, el responsable de la entrega deberá etiquetar en el GIT con el formato **"r vx.x.x"**. Al momento de realizar el pedido de instalación por la vía formal establecida por la ASI (actualmente es la herramienta Mantis) debe informarse la última etiqueta con formato de RELEASE.

Si una versión aprobada llegara a producción y la misma contuviese errores, el Referente de proyecto en conjunto con el Equipo de Desarrollo (Interno o Proveedor) deberán corregir dichos errores, quitar la etiqueta de release de ese commit, agregar al mismo la etiqueta de "**rf vx.x.x**" y crear un nueva etiqueta release en el commit donde los mismos estuvieran corregidos. Hecho esto se deberá enviar la versión generada nuevamente con su correspondiente pedido.

Es obligatorio contar en la carpeta ***scripts,*** con los comandos necesarios para efectuar un rollback en caso de que el deploy puediera fallar. Este script deberá efectuar tanto rollback de código como de base de datos y configuración. Es responsabilidad del desarrollador mantener la lógica necesaria para evitar que se pierda información valiosa en este proceso. Todos los script deben contar con una forma adicional que verifica y valida que la ejecución es exitosa o no, a modo de validación como por ejemplo que contenga las instrucciones que logueen en un log que pueda ser rescatado y enviado a los interlocutores.
Adicionalemente debe ser utilizado el comando de mayor nivel de información(verbouse)

En caso de existir integraciones/acoplamiento con servicios externos al paquete a instalar debe contarse con una forma de verificar que la instalación relacionada se comunica correcta o

incorrectamente y es obligatorio detallar la forma de integración en el documento de arquitectura.

Los ambientes donde se implementan las entregas no contemplan accesos a servicios externos, por lo tanto cualquier excepción debe estar documentada, justificada y la ASI puede aprobar o no el acceso. Para no generar reworking solicitamos anticipar este tipo de necesidad al comienzo del proyecto.

Si se considera que el deploy a realizar puede poner en riesgo la información almacenada en la base de datos del proyecto se deberá indicar al área de infraestructura la realización de un backup de los datos, previo a la realización del mismo, en el ***README.md*** y en la solicitud del pedido del pasaje de ambiente.

Debe existir una rama estable ***(master)*** que es la que siempre contiene las versiones entregadas a la ASI y se recomienda generar una rama de desarrollo ***(dev)*** para desarrollo.

En el archivo ***.gitignore*** dentro del repositorio deberán incluirse los archivos de configuración y todos aquellos archivos que no formen parte del proyecto y que por alguna razón existan en el directorio del mismo (Ej.: archivos de sistema, configuración, archivos subidos por usuarios, etc.).
En la carpeta ***Tests/*** incluir los casos de pruebas funcionales y no funcionales y los informes con la evidencia de los resultados.

El archivo de dependencias deberá estar incluído dentro de la carpeta source con todas las dependencias externas necesarias para el correcto funcionamiento de la aplicación. A continuación se eumeran los ejemplos para las plataformas reguladas por la ASI:

- requierements.txt para python
- package.json para nodejs
- pom.xml para java
- composer.js para php

### Alta de Usuario

En caso de no tener un usuario de la herramienta, el Referente técnico de la ASI realizará las gestiones necesarias para brindarle uno con los permisos necesarios así como la creación de la estructura de carpeta correspondiente.

### CONFIGURACION

A continuación una guía de los pasos necesarios para vincular un puesto al GIT GCBA:

1. Acceder a la URL http://git-asi.buenosaires.gob.ar/help/ssh para generar la Key de ssh
2. Una vez que se cuenta con la Key, agregarla en el profile del usuario del GIT.
   http://git-asi.buenosaires.gob.ar/profile/keys
3. Ejecutar los comandos para clonar el proyecto que la ASI creó para tal fin e informó
4. Subir los archivos respetando el contenido en las carpetas definidas.