**28/sep/2018**

**DIS-017 Modelo de Operación de Verificación Post-Producción**

**E2-DIS-017 Documento de Políticas de Almacenamiento en Repositorio Institucional**

**Contenido**

[1. Objetivo 3](#_Toc525557062)

[2. Introducción 3](#_Toc525557063)

[2.1. Definiciones, Abreviaciones y Referencias 3](#_Toc525557064)

[3. Alcance 3](#_Toc525557065)

[4. Detalle del Entregable 4](#_Toc525557066)

[4.1. Estructura general 4](#_Toc525557067)

[4.1.1. Estructura en *trunk* y *branches* 4](#_Toc525557068)

[4.2. Nombramiento 8](#_Toc525557069)

[4.2.1. Para *branches* 8](#_Toc525557070)

[4.3. Uso del SVN Server 9](#_Toc525557071)

[4.3.1. Creación de usuarios 9](#_Toc525557072)

[4.3.2. Permisos 9](#_Toc525557073)

[4.3.3. Contraseñas 9](#_Toc525557074)

[4.3.4. Grupos 10](#_Toc525557075)

[5. Conclusiones 10](#_Toc525557076)

[6. Firmas 12](#_Toc525557077)

1. Objetivo

Establecer los lineamientos de operación del repositorio Institucional:

* Estructura organizacional del Repositorio Institucional.
* Políticas de almacenamiento.
* Políticas de creación de *Branch*.
* Reglas de nombrado de archivos.
* Reglas para creación de usuarios.
* Reglas para creación de grupos.
* Reglas para creación de contraseñas.
* Reglas de administración de Permisos.

La aplicación de los lineamientos antes mencionados es de gran relevancia ya que permiten una administración óptima del contenido almacenado en Repositorio Institucional.

1. Introducción

Actualmente el INFONAVIT cuenta con un repositorio donde se encuentra el código fuente de los diversos aplicativos del Instituto. Dicho repositorio requiere de una serie de políticas y lineamientos para su correcta administración.

Actualmente la administración del repositorio se lleva a cabo a través de tres principales directorios de código: *Trunk*, donde se aloja el código de los aplicativos en producción; *Branch,* que representa una ramificación del código en producción, facilitando la implementación de cambios, permitiendo así su manejo sin restricciones, y sin afectación a los aplicativos de producción; y el *Tag*, cuya aplicación sirve para “congelar” el código, impidiendo su manipulación y permitiendo almacenar versiones históricas.

* 1. Definiciones, Abreviaciones y Referencias

| Términos/Siglas | Descripción |
| --- | --- |
| Política | Orientaciones o directrices que rigen la actuación de una persona o entidad en un asunto o campo determinado. |
| Código fuente | Conjunto de líneas de texto, escritas en un lenguaje de programación, cuyo propósito es fungir como directrices que la computadora debe seguir para realizar dicho programa, Dicho lenguaje debe ser procesado para obtener un producto de software ejecutable por la computadora. |
| Lenguaje de Programación | Es un lenguaje formal que especifica una serie de instrucciones para que una computadora realice ciertas acciones o tareas. Los lenguajes de programación pueden usarse para crear programas que pongan en práctica algoritmos específicos que controlen el comportamiento físico y lógico de una computadora. |
| Archivo fuente | Es el archivo producto del conjunto de líneas de código. Dicho archivo debe ser procesado para su posterior ejecución. |
| Versionamiento | Acción de asignar una versión a un determinado objeto. En este contexto, el versionamiento representan todas aquellas ediciones que se realizan a un determinado código y/o aplicativo del Instituto. |

1. Alcance

El presente documento cubre las siguientes características de políticas de almacenamiento en el repositorio:

* Nomenclatura del nombre de los directorios.
* Manejo de repositorio y sus directorios: *Trunk y Branch.*
* Políticas de uso de repositorio.
* Nomenclatura y guía de contraseñas.
* Flujo de trabajo principal, flujos alternos y excepciones.

1. Detalle del Entregable

A continuación, se definen los distintos tipos de políticas a aplicar para la correcta administración del repositorio institucional:

* 1. Estructura general

La estructura general del repositorio se compone de 3 principales directorio: *Trunk*, *Branch y Tag*

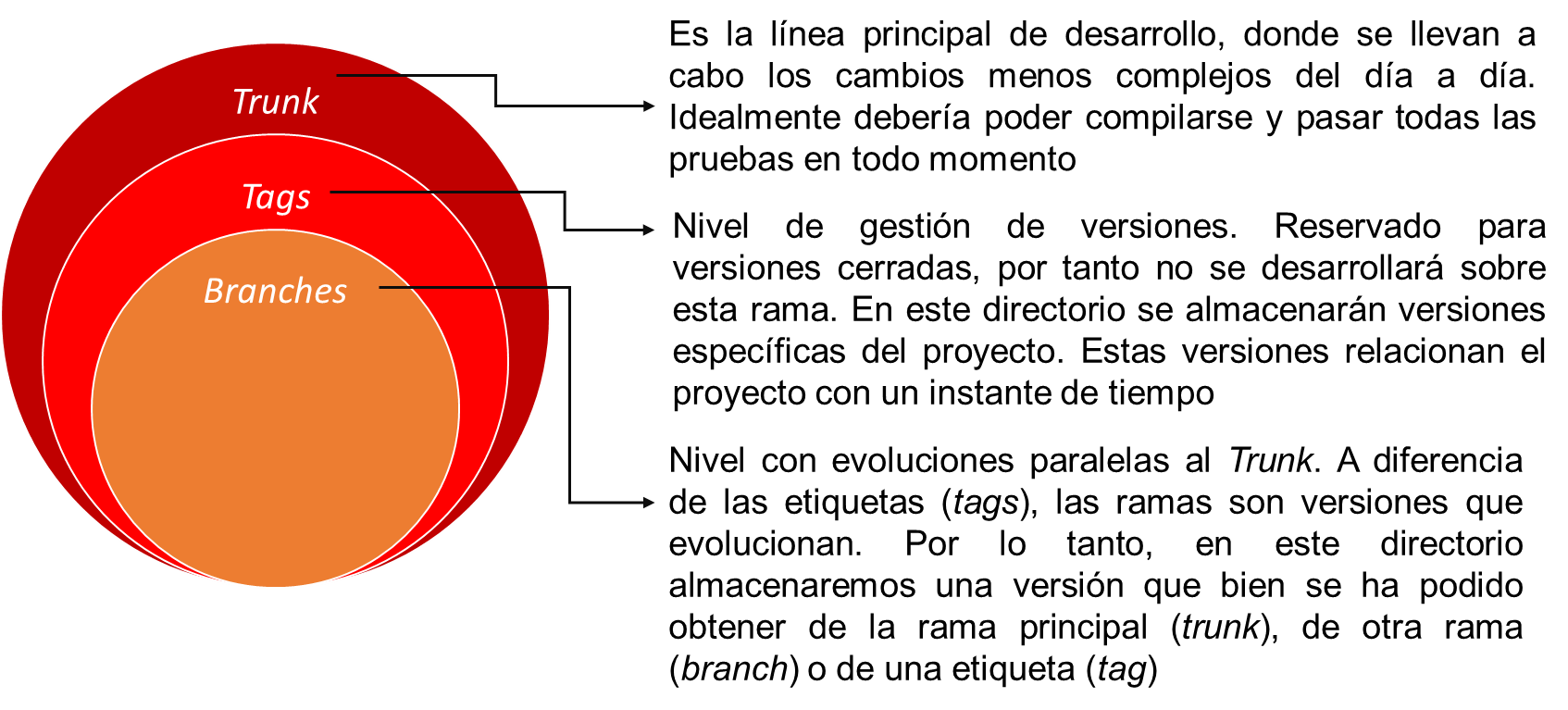


Ilustración 1 Estructura Principal

Estos directorios contienen las siguientes carpetas:

* **Código**: “*Workspace*” del proyecto con el código fuente del aplicativo.
* **Instalable**: Archivo generado por medio del procesamiento del código fuente.
* **Documentación**: Documentación generada para el proyecto.
* **Base de datos**: Se subdivide en:
  + ***DbSchema***: Estructura donde se alojan las bases de datos.
  + ***Shell***: Instrucciones textuales para la creación y administración de la base de datos.
  + ***StoredProcedure***: Procedimientos o instrucciones programadas que realiza el manejador de base de datos.
  + ***Scripts***: Código de operaciones de administración de base de datos.
    1. Estructura en *trunk* y *branches*

Además de la estructura especificada anteriormente, en la carpeta **instalable** de los directorios *trunk* y *branches* estarán contenidos dos archivos instalables: uno *con código fuente*, y otro *sin código fuente*, agregando en el nombramiento del instalable los sufijos ‘*\_ccf’* y *‘\_scf’*, respectivamente.

**Ej: MonitorInscripcion\_ccf.ear y MonitorInscripcion\_scf.ear**

***Nota:*** *Como tal, no existe nomenclatura alguna para el nombramiento de los instalables. Sin embargo, se recomienda que su nombre aluda al aplicativo del instituto y sea de fácil comprensión.*

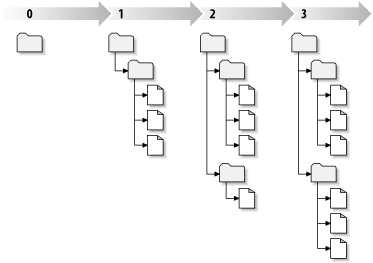


Ilustración 2 Carpetas

***Nota:*** *Sólo se crearán los directorios necesarios para el requerimiento (e.g. si un branch requiere modificaciones en código y documentación, entonces sólo se crearán o modificarán esos dos directorios).*

* + 1. **Aseguramiento del correcto despliegue**

Debido a la naturaleza cambiante de los aplicativos y su desarrollo, se debe asegurar que el código que surge de los *branches* y el *trunk* sea desplegado correctamente en producción, y no sufra cambio alguno. Para ello se utilizará la funciónresumenMD5*.*

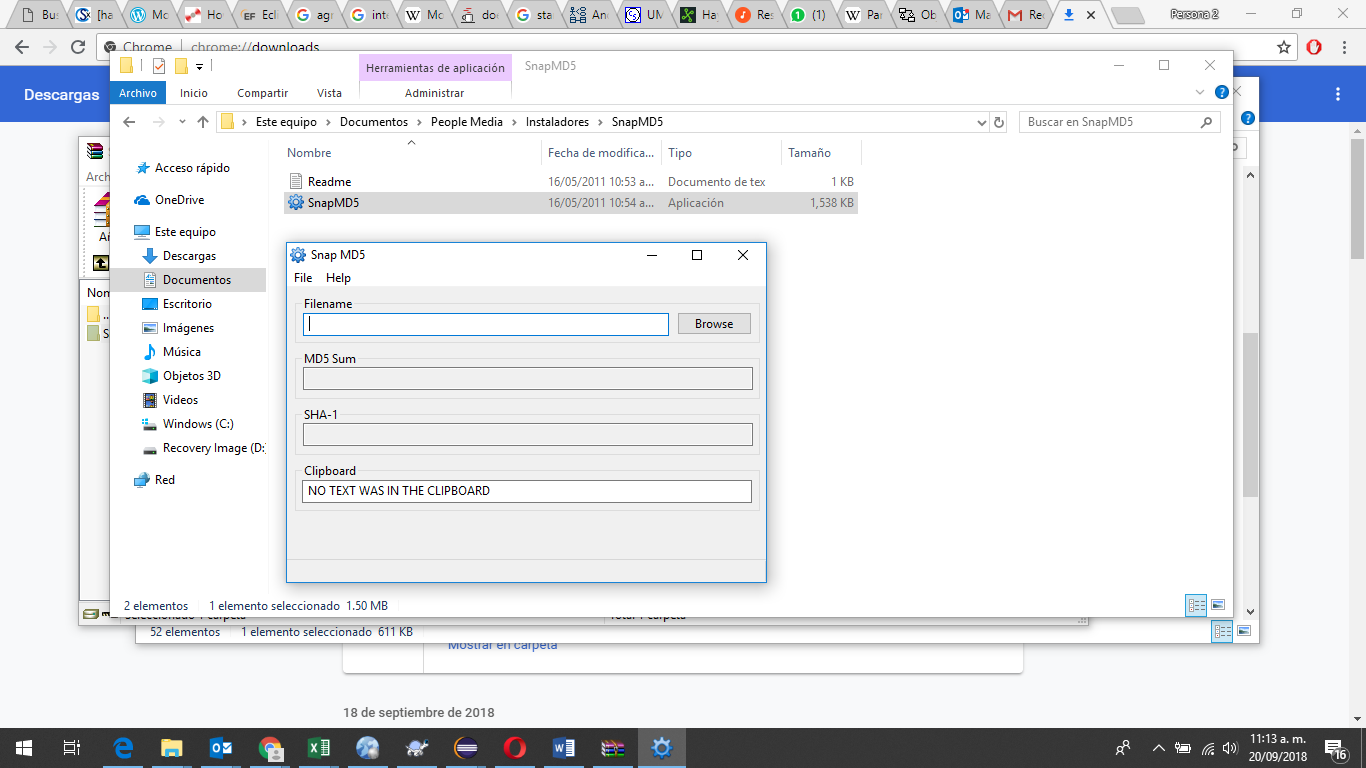
La función MD5 es ampliamente usada en el ámbito de seguridad de la información y en los procesos de cifrado de la misma. Funciona calculando una clave alfanumérica única de un archivo o directorio comprimido. Para ello utiliza un *algoritmo de resumen de mensaje*. Esta clave es altamente sensible a los cambios; es decir, si se realiza la más ínfima modificación al archivo (e. g. un salto de línea en el código, o cambio de nombre a una variable), la clave MD5 cambiará.

De esta manera, se debe calcular la clave MD5 de los instalables con y sin código justo antes de su puesta en producción. Una vez en producción, se deben calcular las claves de ambos instalables nuevamente, y compararlas con las existentes en el *branch* y *trunk* correspondientes.

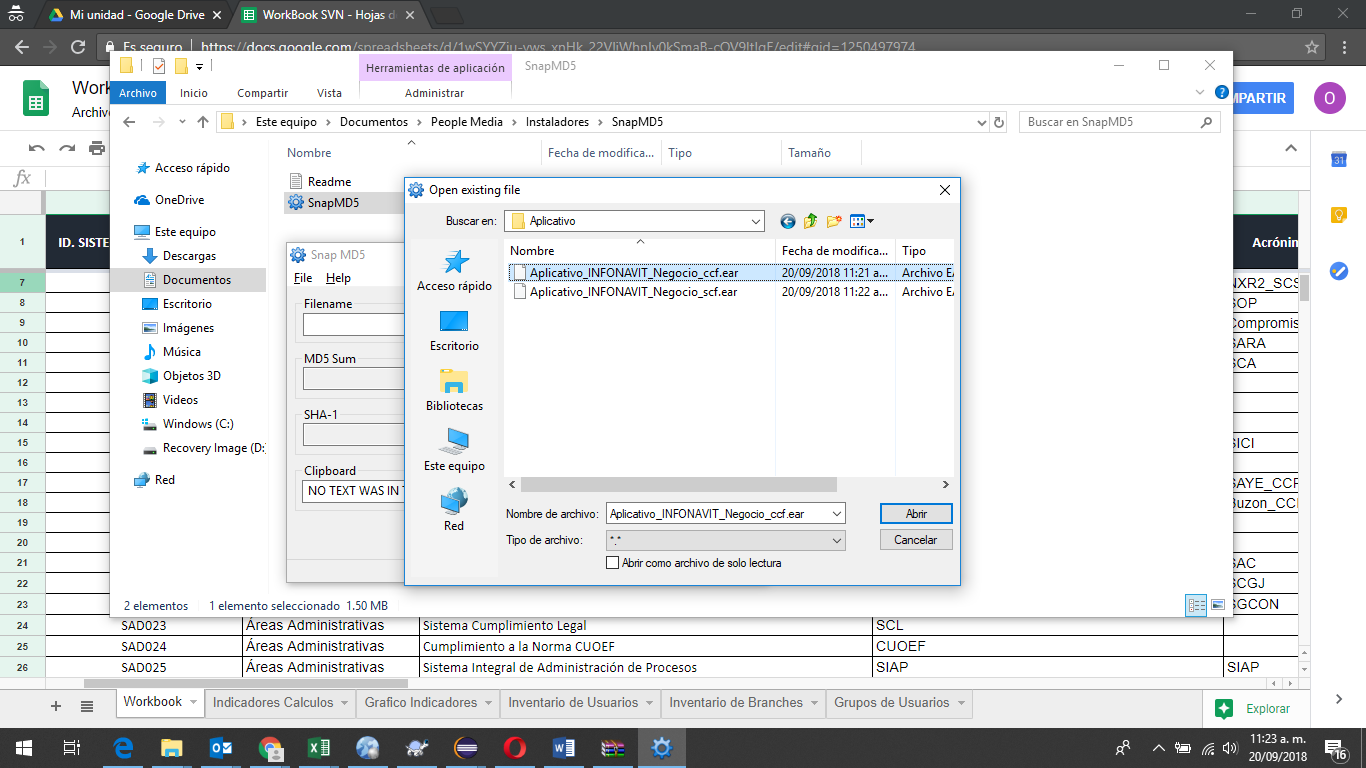
* + - 1. **Clave MD5**

El procedimiento para el cálculo de la clave MD5 es el siguiente:

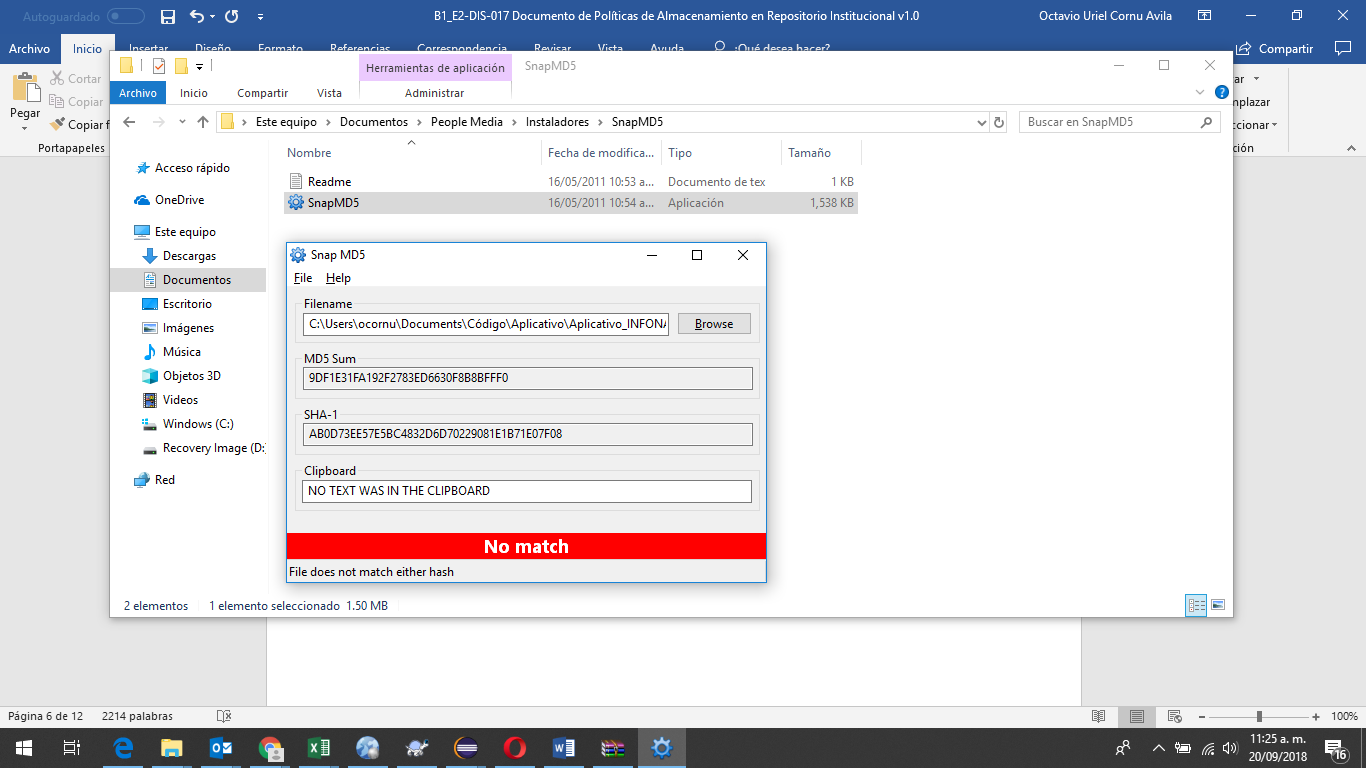
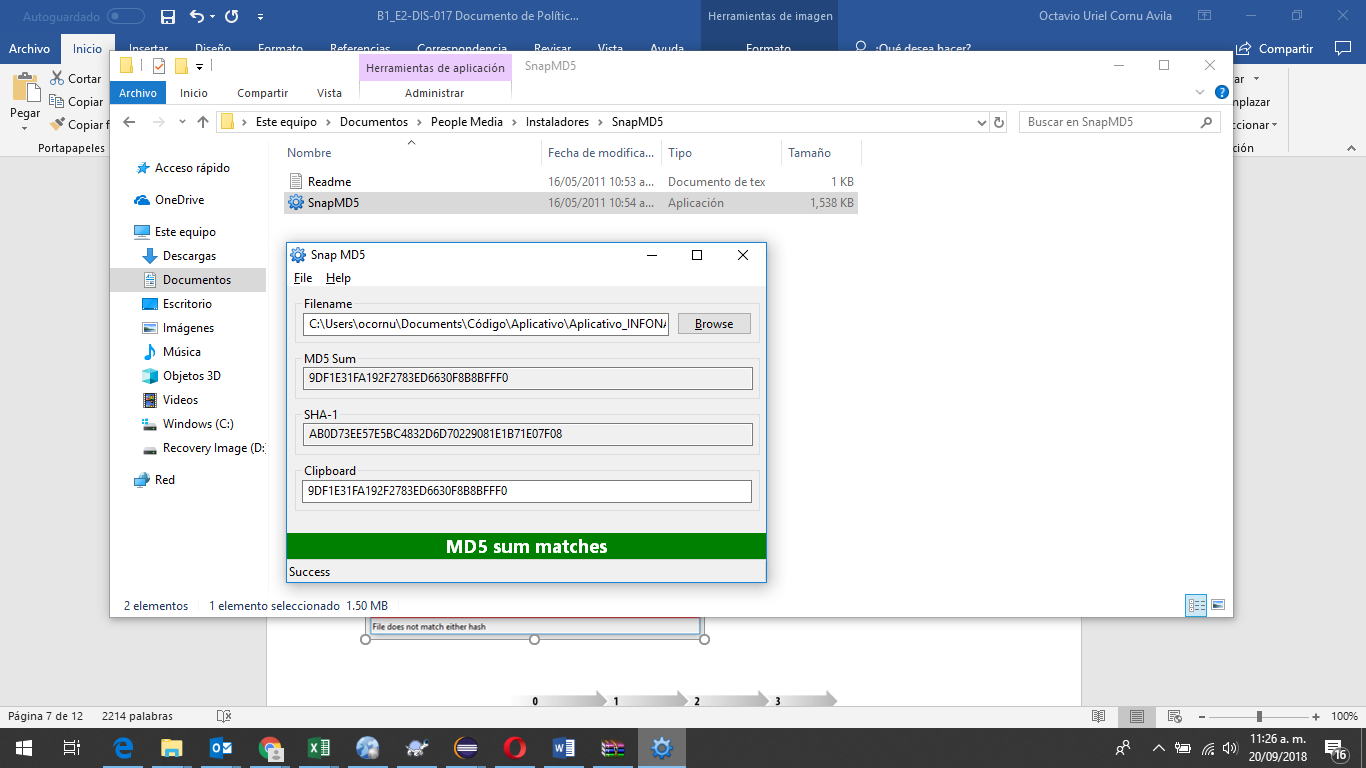
1. Descargar los archivos instalables del aplicativo, con y sin código fuente, desde el directorio *Instalable*, del *branch*, en el repositorio *SVN Server*.
2. Descargar, descomprimir y ejecutar la herramienta *open source* SnapMD5, desde: <https://www.filecroco.com/download-snap-md5/download/>



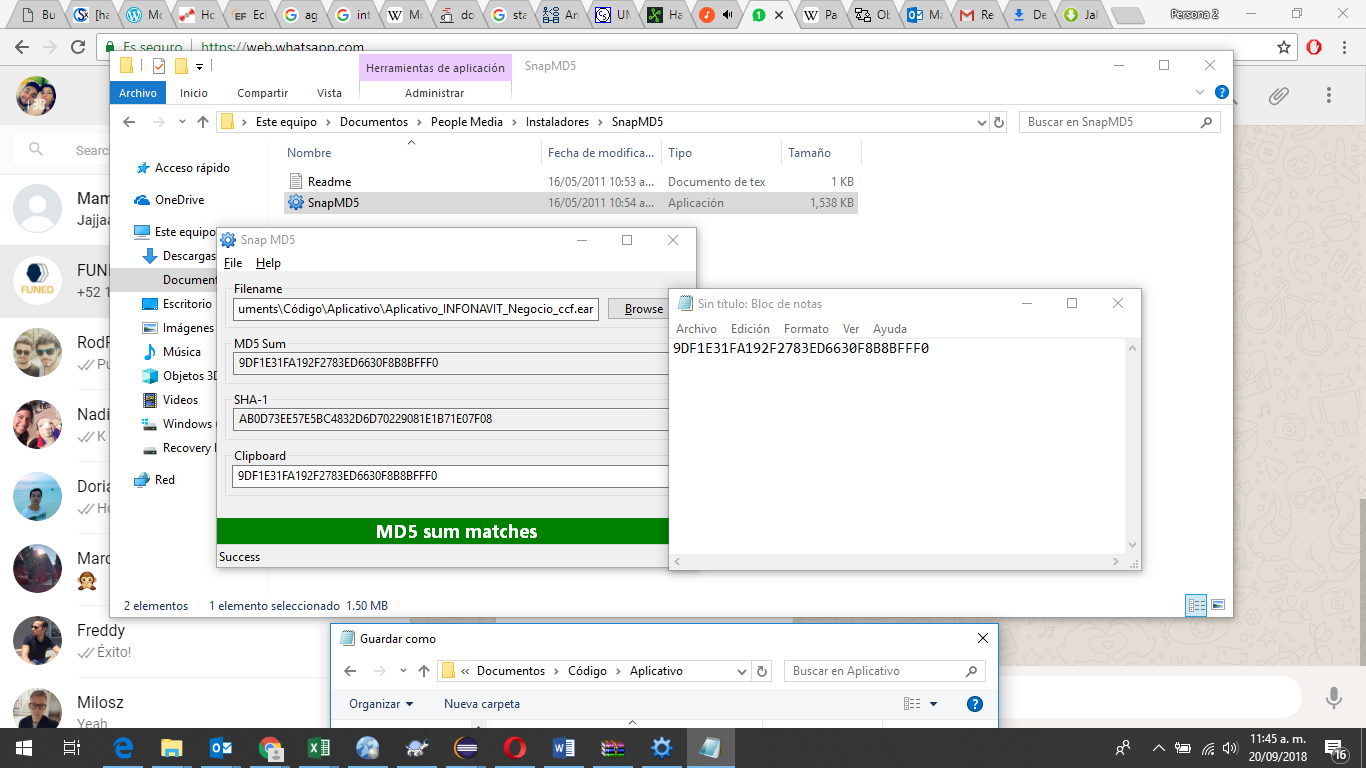
1. Buscar y seleccionar el archivo instalable con código fuente, usando el botón *browse*.

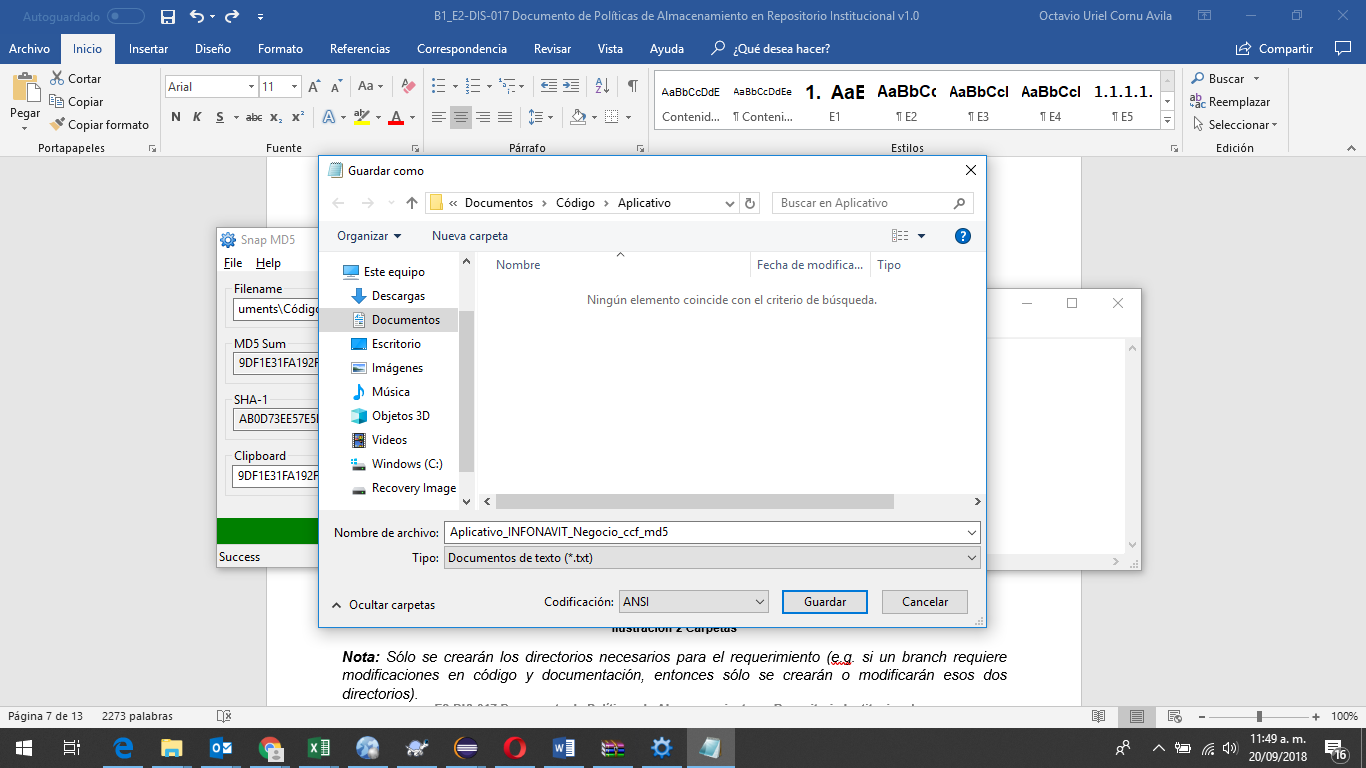


1. Copiar el contenido del apartado ‘*MD5 sum’* y pegar en el apartado ‘*Clipboard’.*

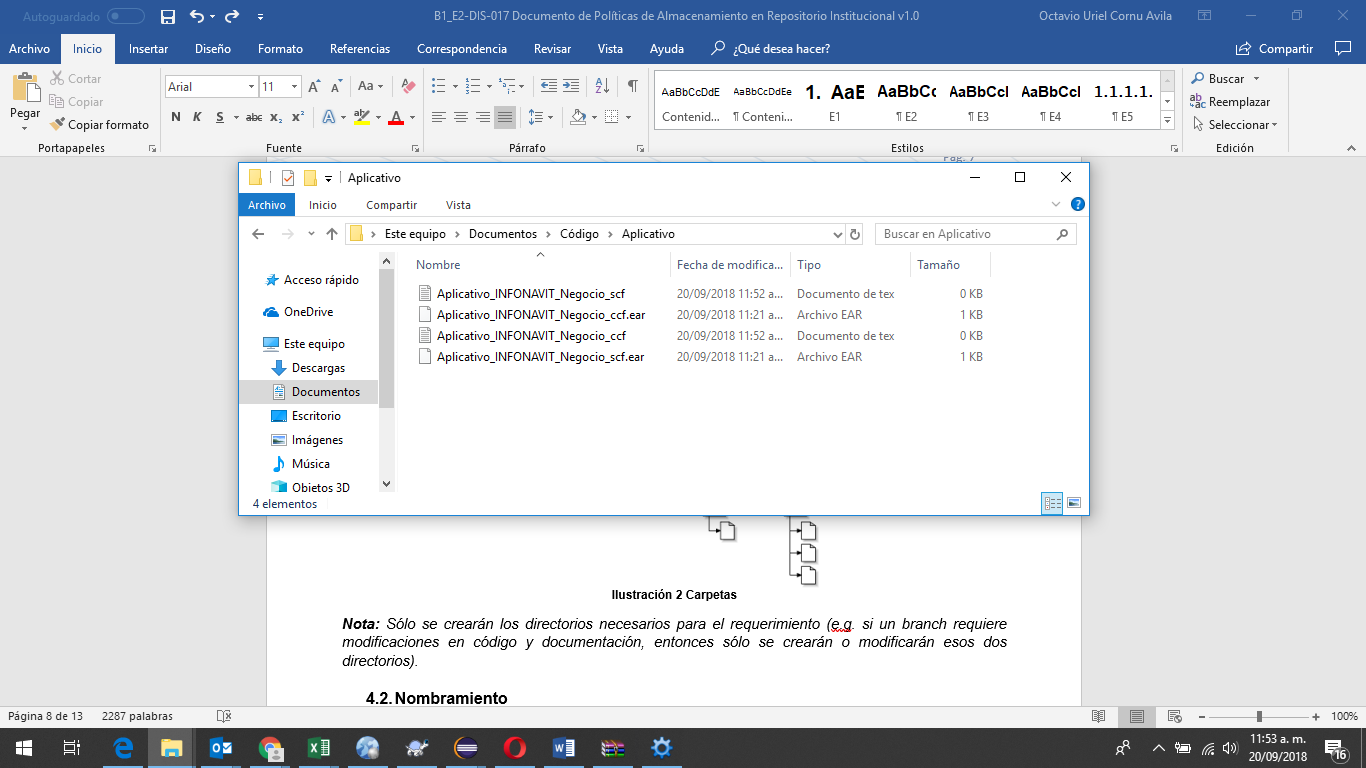
 

1. A su vez, copiar el contenido de la clave MD5 y pegarlo en un archivo ‘.txt’, bajo el mismo nombre del aplicativo, agregando el sufijo ‘\_md5’.





1. Repetir los pasos 3 al 5 para el archivo instalable sin código fuente.



1. Subir al directorio *branch* del aplicativo los archivos instalables y los archivos de texto.
2. El equipo de liberación deberá ejecutar los pasos 1 al 6, y comparar las nuevas claves MD5 con las contenidas en el *branch*, pegando las claves de los archivos de texto del *branch,* en el apartado ‘*Clipboard*’ de la herramienta Snap MD5, previa puesta en producción.
3. Una vez en producción, se deberá calcular la clave MD5 de ambos archivos, misma que deberá coincidir con las claves calculadas por el equipo de liberación y las contenidas en el *branch*, asegurando así que los archivos instalables en producción no sufrieron cambios que podrían afectar la operación del Instituto.
   1. Nombramiento
      1. Para *branches*

Los nombres de los directorios seguirán la siguiente estructura:

* **br\_ACRONIMO\_DE\_LA\_APLICACION\_XXXXXXXX**, donde “XXXXXXXX” es el código de requerimiento registrado en Jira, y puede estar conformado por letras, números, o una combinación de ambas (e.g. br\_SARA\_PLASGYJ\_6).

En el supuesto que no se cuente con un número de requerimiento, debe utilizarse un identificador genérico, el cual debe poseer la siguiente estructura: **REQUEXXXXXXN**. Desglosándose de la siguiente manera:

* + **REQUE:** Valor fijo que señala que se trata de un requerimiento.
  + **XXXXXX:** Números que representan la fecha de creación del *branch* en formato DDMMAA.
  + **N:** Número de requerimiento del día. Es consecutivo y depende de la cantidad de requerimientos sin código asignado que se generen en ese día.

***Nota****: Si un directorio del branch requiere cambio de nombre, derivado de la asignación de un requerimiento posterior o cualquier otro motivo, deberá notificarse a los usuarios que trabajen en dicho directorio ya que se modificará la URL del branch.*

* 1. Uso del SVN Server

Para el uso del servidor SVN, así como la administración de sus usuarios, tenemos las siguientes políticas:

* + 1. Creación de usuarios

Para la creación de los accesos de usuarios, se recomienda estructurarlos en la forma **primernombre.primerapellido.** Por ejemplo, para un usuario de nombre “Pedro Pérez Salinas” su acceso sería: *pedro.perez*

* Si existe más de un usuario con los mismos primeros nombres y apellidos, **se agregará al acceso del nuevo usuario, al final del primer nombre, la inicial del segundo nombre.** Por ejemplo, si un usuario se llama Pedro Alejandro Pérez Ulloa, su acceso sería *pedroa.perez*.

Si el usuario no tiene segundo nombre, se agregará la inicial del segundo apellido al final del primer apellido. Siguiendo el mismo ejemplo, el acceso sería el siguiente: pedro.perezu

* Si existe más de un usuario cuyos primeros nombres, primeros apellidos, e iniciales coincidan a la perfección, entonces se agregará un número al final del acceso. Por ejemplo, si un usuario se llama Pedro Antonio Pérez Urales, su acceso sería *pedroa.perezu1*. Esta numeración será en función de la cantidad de nuevos usuarios coincidentes.
  + 1. Permisos

El servidor Visual SVN posee tres niveles de permisos, otorgables a un usuario o grupo:

* Sin acceso
* Acceso solo lectura
* Acceso lectura-escritura

De igual manera, al finalizar la atención de la solicitud se modificarán los accesos otorgados al usuario a “Sin acceso”. Dicha modificación permitirá al usuario ingresar al repositorio SVN, pero sin la posibilidad de consultar su contenido. En caso de requerir una modificación sobre la nueva línea base del código, es necesario generar una nueva solicitud.

* + 1. Contraseñas

Se sugiere a los usuarios utilizar el tipo de estructura *leet*, que consiste en reemplazar ciertas letras por números o por un conjunto de caracteres especiales. Por ejemplo. La letra ‘E’ por el número 3, o la letra ‘G’ por ‘(\_>’, etc. Adicionalmente, se agregan las siguientes bases para reforzar la creación de las contraseñas de usuario:

1. La contraseña deberá estar en mayúsculas.
2. Se recomienda que su longitud sea entre 4 y 6 caracteres. En caso de utilizar *leet*, cada letra será considerada como un carácter. Por ejemplo, en *leet* podemos escribir la ‘D’ como ‘|)’, técnicamente está conformada por dos caracteres, pero para fines de creación de contraseñas se considerará como uno solo.
3. Se recomienda generar la contraseña usando el alfabeto *leet*, como una combinación de las dos letras iniciales del primer nombre y las dos iniciales del primer apellido, para el caso de contraseña de 4 caracteres; y para las de 6 caracteres, agregar las dos últimas del primer apellido. Por ejemplo, la contraseña de Pedro Alejandro Pérez Ulloa, utilizando estructura *leet* sería: |D3/-\|\_; correspondiendo ‘|D’ a la letra ‘P’; ‘3’ a la letra ‘E’; ‘/-\’ a la letra ‘A’; y ‘|\_’ a la letra ‘L’.
4. El alfabeto *leet* permite sustituir las letras del alfabeto por números o caracteres de varias formas recomendadas, las cuales se muestran a continuación:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Letra** | **Equivalente en alfabeto *leet*** |  | **Letra** | **Equivalente en alfabeto *leet*** |
| **A** | 4, /-\, @, ^, /\ , //-\\ |  | **N** | /\/, |\|, (\), /|/, [\], {\}, ][\][, []\[] |
| **B** | 8, ]3, ]8, |3, |8, ]]3 |  | **O** | 0, (), <>, \*, [[]] |
| **C** | (, { , [[ |  | **P** | |D, |\*, |>, []D, ][D |
| **D** | ), [}, |), |}, |>, [>, ]]) |  | **Q** | (,), 0, O, (comas necesarias), [\] |
| **E** | 3, ii |  | **R** | |2, |?, |-, ]]2 []2 ][2 |
| **F** | |=,(=, ]]= |  | **S** | 5, $ |
| **G** | 6, 9, (\_>, [[6 |  | **T** | 7, +, ']', 7`, ~|~, -|-, '][' |
| **H** | #, |-|, (-), )-(, }{, }-{, {-}, /-/, \-\, |~|, []-[], ]]-[[ |  | **U** | (\_), |\_|, \\_\, /\_/, \\_/, []\_[], ]\_[ |
| **I** | 1, !, |, ][, [] |  | **V** | \/ , \\// |
| **J** | \_|, u|, ;\_[], ;\_[[ |  | **W** | \/\/, |/\|, [/\], (/\), VV, ///, \^/, \\/\// |
| **K** | |<, |{, ][<, ]]<, []< |  | **X** | ><, }{, )( |
| **L** | |,1, |\_, []\_, ][\_ |  | **Y** | '/, %, `/, \j , ``// |
| **M** | /\/\, |\/|, [\/], (\/), /V\, []V[], \\\, (T), ^^, .\\, //., ][\\//][ |  | **Z** | 2, z, 7\_ |

1. Al elegir la contraseña, se pueden sustituir únicamente las vocales o el total de las letras. Por ejemplo:
   * Para un usuario de nombre Pedro Pérez:
     1. Modo 4 letras solo vocales: P3P3
     2. Modo 6 letras solo vocales: P3P33Z
     3. Modo 4 letras todas las letras: |>3|>3
     4. Modo 6 letras todas las letras: |>3|>337\_
     5. Grupos

Los grupos se crean para facilitar la administración de los usuarios que tengan o requieran el mismo perfil. La nomenclatura para la creación de los grupos debe ser la siguiente:

* Todo grupo debe iniciar con las iniciales “INFON” seguidos del separador “-”, quedando de la siguiente forma: “INFON-”
* El identificador debe ir inmediatamente después del separador y contendrá las iniciales del departamento o área que está solicitado los accesos. Por ejemplo, para un grupo donde estarán los usuarios del departamento de ‘Dependencia Técnica’, le correspondería el siguiente nombre de grupo: INFON-DT.
* Existe la posibilidad que un grupo incluya a otro u otros usuarios y/o grupos, si el Instituto así lo requiere.
* La administración de los distintos grupos estará a cargo y discreción del equipo de *Configuration Management*.
* Los usuarios que finalicen su interacción con el repositorio serán agregados al grupo ‘INFON-INACTIVO’, con el objetivo de remover los permisos sobre los directorios con los que interactuaron.

1. Conclusiones

Las políticas de almacenamiento en repositorio nos marcan los lineamientos del Instituto para la correcta administración de los códigos fuente de los aplicativos del mismo. Dichos lineamientos facilitan la documentación de los aplicativos mediante la administración de permisos, ya que mantenemos un registro de versionamiento y cambios, a la par que marcamos la trazabilidad de los mismos con sus responsables.

A su vez, la estructura del *SVN* *Server* realiza un manejo óptimo de incidencias, ya que su estructura evita, en la medida de lo posible, que implementaciones del código susceptibles a errores y/o vulnerabilidades sean puestas en ambiente productivo, evitando así la afectación de la operación del Instituto.

La administración de los usuarios en el servidor, a la par de sus permisos, nos dan la facultad de registrar los cambios que realicen los usuarios o grupos de usuarios, además de asegurar que sólo los interesados tengan acceso al código y realicen cambios en el mismo.

1. Firmas

Por parte de INFONAVIT:

|  |  |  |
| --- | --- | --- |
| **Supervisó** |  | **Autorizó** |
|  |  |  |
|  |  |
|  |  |
|  |  |
| Héctor Campos Campos |  | Alejandro De León Languré |
| **Gerente de Ingeniería de TI** |  | **Gerente Sr. de Arquitectura e Ingeniería de TI** |

Y por parte de People Media S.A de C.V.

|  |  |  |
| --- | --- | --- |
| **Elaboró** |  | **Revisó** |
|  |  |  |
|  |  |
|  |  |
|  |  |
| Hernán J. Cortés Yip |  | Ing. Alexis Villa Reyes |
| **Consultor de TI** |  | **Gerente de Proyecto** |