Repositorios.

Un repositorio, depósito o archivo es un sitio centralizado donde se almacena y mantiene información digital, habitualmente bases de datos o archivos informáticos. ( significa armario o alacena)

Para recopilar, gestionar, difundir y preservar la producción documental de la institución, cualquiera que sea su tipología, a través de la creación de una colección digital organizada, abierta e interoperable para garantizar impacto y visibilidad.

Características generales

Los datos almacenados en un repositorio pueden distribuirse a través de una [red informática](http://es.wikipedia.org/wiki/Red_inform%C3%A1tica), como [Internet](http://es.wikipedia.org/wiki/Internet), o de un medio físico, como un [disco compacto](http://es.wikipedia.org/wiki/Disco_compacto). Pueden ser de acceso público o estar protegidos y necesitar de una autentificación previa. Los repositorios más conocidos son los de carácter académico e institucional. Los repositorios suelen contar con [sistemas de respaldo](http://es.wikipedia.org/wiki/Copia_de_seguridad) y mantenimiento preventivo y correctivo, lo que hace que la información se pueda recuperar en el caso que la máquina quede inutilizable. A esto se lo conoce como preservación digital,[1](http://es.wikipedia.org/wiki/Repositorio#cite_note-1) y requiere un exhaustivo trabajo de control de calidad e integridad para realizarse correctamente.

Depositar no debe confundirse con publicar. El depósito en los repositorios es una manera de comunicar públicamente los trabajos de los investigadores, aumentando su difusión: los autores ponen disponibles en acceso abierto una versión de los artículos que han publicado en revistas, tradicionales o de acceso abierto.[[cita requerida](http://es.wikipedia.org/wiki/Wikipedia:Verificabilidad)] Para ello, los sistemas de repositorios suelen integrarse e interoperar con otros sistemas y aplicaciones web. Asimismo, los repositorios cumplen un rol importante en la formación universitaria.

Algunas instituciones promueven el uso de sus repositorios como un servicio adicional para el investigador. Otras instituciones poseen mandatos propios que obligan a los autores o investigadores a depositar sus publicaciones (o determinados tipos, como por ej. tesis doctorales) en el repositorio institucional, con fines de visibilidad, impacto y preservación.[6](http://es.wikipedia.org/wiki/Repositorio#cite_note-6)En algunos países, como por ejemplo [Argentina](http://es.wikipedia.org/wiki/Argentina), se han promulgado leyes de acceso abierto que promueven la implementación y uso de los repositorios de instituciones sustentadas con fondos públicos,[7](http://es.wikipedia.org/wiki/Repositorio#cite_note-7) mientras que otros países están trabajando en la aprobación de leyes similares, como por ejemplo México.[8](http://es.wikipedia.org/wiki/Repositorio#cite_note-8)

|  |  |
| --- | --- |
| Básicamente, son sitios en los cuales se alojan objetos digitales como vídeos, animaciones, imágenes, documentos, libros, entre otras cosas. Asimismo, son organizados por categorías, áreas de conocimiento, temas y tipo, por mencionar algunos. Su objetivo es facilitar la búsqueda, acceso y visualización del material digital, así como la preservación y difusión de materiales. | |
|  |  |
| ¿Dónde encontrar repositorios de objetos digitales?  Los repositorios se encuentran disponibles en la web y existen listados realizados por algunas universidades u organizaciones. Los más representativos se encuentran en el directorio [ROAR](http://roar.eprints.org/), [OpenDOAR](http://www.opendoar.org/), en la lista de [repositorios de la Universidad de Illinois](http://www.ideals.illinois.edu/).   En el 2011 estos fueron los repositorios considerados como los mejores del mundo:   * [Social Science Research Network](http://ssrn.com/) * [Arxiv.org e-Print Archive](http://arxiv.org/) * [CiteSeerX](http://citeseerx.ist.psu.edu/) * [Research Papers in Economics](http://repec.org/) * [Smithsonian/NASA Astrophysics Data System](http://adsabs.harvard.edu/) | |
|  | |

Subversión

El término subversión (del [latín](http://es.wikipedia.org/wiki/Lat%C3%ADn) subvertere: trastocar, dar vuelta) , se refiere a un proceso por el que los valores y principios de un sistema establecido, se invierten, y se relaciona con un [trastorno](http://es.wikipedia.org/wiki/Enfermedad), una [revuelta](http://es.wikipedia.org/wiki/Revuelta) o una [destrucción](http://es.wikipedia.org/wiki/Destrucci%C3%B3n); está presente en todos los [idiomas](http://es.wikipedia.org/wiki/Idioma) de origen latino, originalmente aplicado a diversos eventos como la derrota militar de una [ciudad](http://es.wikipedia.org/wiki/Ciudad).

Subversión (SVN) es una herramienta de control de versiones open source basada en un repositorio cuyo funcionamiento se asemeja enormemente al de un sistema de ficheros. Es software libre bajo una licencia de tipo Apache/BSD.

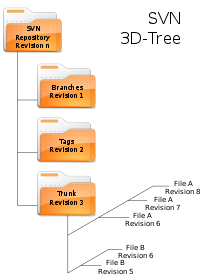
Características generales

Permite al usuario crear, copiar y borrar carpetas con la misma flexibilidad con la que lo haría si estuviese en su disco duro local. Dada su flexibilidad, es necesaria la aplicación de buenas prácticas para llevar a cabo una correcta gestión de las versiones del software generado.

Subversión puede acceder al repositorio a través de redes, lo que le permite ser usado por personas que se encuentran en distintas [computadoras](http://es.wikipedia.org/wiki/Computadora). A cierto nivel, la posibilidad de que varias personas puedan modificar y administrar el mismo conjunto de datos desde sus respectivas ubicaciones fomenta la colaboración. Se puede progresar más rápidamente sin un único conducto por el cual deban pasar todas las modificaciones. Y puesto que el trabajo se encuentra bajo el control de versiones, no hay razón para temer por que la calidad del mismo vaya a verse afectada —si se ha hecho un cambio incorrecto a los datos, simplemente deshaga ese cambio.[2](http://es.wikipedia.org/wiki/Subversion_(software)#cite_note-2)

Ventajas

* Se sigue la historia de los archivos y directorios a través de copias y renombrados
* Las modificaciones (incluyendo cambios a varios archivos) son [atómicas](http://es.wikipedia.org/wiki/Atomicidad).
* La creación de ramas y etiquetas es una operación más eficiente. Tiene coste de complejidad constante (O(1)) y no lineal (O(n)) como en CVS.
* Se envían sólo las diferencias en ambas direcciones (en CVS siempre se envían al servidor archivos completos).
* Puede ser servido mediante [Apache](http://es.wikipedia.org/wiki/Servidor_HTTP_Apache), sobre [WebDAV](http://es.wikipedia.org/wiki/WebDAV" \o "WebDAV)/[DeltaV](http://es.wikipedia.org/w/index.php?title=DeltaV&action=edit&redlink=1" \o "DeltaV (aún no redactado)). Esto permite que clientes WebDAV utilicen Subversión de forma transparente.
* Maneja eficientemente archivos binarios (a diferencia de CVS que los trata internamente como si fueran de texto).
* Permite selectivamente el bloqueo de archivos. Se usa en archivos binarios que, al no poder fusionarse fácilmente, conviene que no sean editados por más de una persona a la vez.
* Cuando se usa integrado a Apache permite utilizar todas las opciones que este servidor provee a la hora de autentificar archivos ([SQL](http://es.wikipedia.org/wiki/SQL), [LDAP](http://es.wikipedia.org/wiki/LDAP), [PAM](http://es.wikipedia.org/wiki/Autenticaci%C3%B3n#PAM), etc.).

[](http://commons.wikimedia.org/wiki/File:Svn_3D-tree.svg)

Árbol de proyecto subversión.

Carencias

* El manejo de cambio de nombres de archivos no es completo. Lo maneja como la suma de una operación de copia y una de borrado.
* No resuelve el problema de aplicar repetidamente parches entre ramas, no facilita llevar la cuenta de qué cambios se han realizado. Esto se resuelve siendo cuidadoso con los mensajes de commit.

Uso y reconocimiento

Subversión es muy conocido en la comunidad de software libre y se utiliza en muchos proyectos, como:

[Apache Software Foundation](http://es.wikipedia.org/wiki/Apache_Software_Foundation)

[Django](http://es.wikipedia.org/wiki/Django_(framework)) (migrado a Git)

[Free Pascal](http://es.wikipedia.org/wiki/Free_Pascal)

[FreeBSD](http://es.wikipedia.org/wiki/FreeBSD)

[GCC](http://es.wikipedia.org/wiki/GNU_Compiler_Collection)

[KDE](http://es.wikipedia.org/wiki/KDE) (en proceso de migración a Git)

* [Ruby](http://es.wikipedia.org/wiki/Ruby)
* [Tigris.org](http://es.wikipedia.org/w/index.php?title=Tigris.org&action=edit&redlink=1)
* Servicios que proporcionan almacenamiento usando Subversion gratuito para proyectos de software libre:
* [SourceForge](http://es.wikipedia.org/wiki/SourceForge)
* [Google Code](http://es.wikipedia.org/wiki/Google_Code)
* [Project Kenai](http://es.wikipedia.org/w/index.php?title=Project_Kenai&action=edit&redlink=1)
* [CodePlex](http://es.wikipedia.org/wiki/CodePlex)
* [Forja de Conocimiento Libre de la Comunidad RedIRIS](http://es.wikipedia.org/wiki/Forja_de_Conocimiento_Libre_de_la_Comunidad_RedIRIS) de [RedIRIS](http://es.wikipedia.org/wiki/RedIRIS" \o "RedIRIS)
* Subversión también está siendo adoptado en el mundo corporativo. En un informe 2007 de Forrester Research, reconocía a Subversion como el líder destacado en la categoría de[sistema de control de versiones](http://es.wikipedia.org/wiki/Revision_Control_System).[3](http://es.wikipedia.org/wiki/Subversion_(software)#cite_note-3)

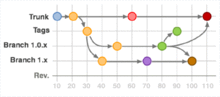
A continuación se listan las funcionalidades que se le debería dar a cada rama del repositorio:

* Trunk: Rama de desarrollo principal.
* Tags: Rama de gestión de versiones. Reservado para versiones cerradas, por tanto no se desarrollará sobre esta rama.
* Branches: Rama con evoluciones paralelas al Trunk.
* Operaciones Habituales con subversión
* A continuación se presentan las operaciones más habituales con las que nos encontramos trabajando con subversión.

Ejemplo de Ciclo de Vida de un proyecto con subversiones, a continuación se explicará paso a paso la transición de estados de un posible software siendo desarrollado usando subversión como herramienta de control de versiones.

Se partirá de un código inicial que se evolucionará hasta cerrar una primera versión. Esta versión se llevará a producción y a partir de ahí se empezará a trabajar sobre la siguiente versión.

Para mostrar distintos escenarios de ramificaciones de código, se supondrá que se contrata un servicio de mantenimiento evolutivo del producto entregado, que tendrá que desarrollarse en paralelo a la evolución de la siguiente versión del producto y que además se detectará un pequeño bug en el producto que requerirá una corrección urgente, y que por tanto no podrá esperar a resolverse en una nueva versión del producto.

[](http://commons.wikimedia.org/wiki/File:SVN_Branches_and_Trunk.gif)

Método propuesto para desarrollar versiones en SVN

1. Rev. 10: Se supone un estado inicial con el código fuente de partida dado de alta en el Trunk. El resto del repositorio queda vacío.
2. Rev. 20: Evolución de la versión 1.0.0 del SW mediante Trabajo en Equipo sobre el Trunk.
3. Rev. 30: Cierre de Versión 1.0.0 del SW. Implica la creación de un Tag y el paso al desarrollo de la versión 2.0.0 en el Trunk. Suponemos la puesta en producción de la versión 1.0.0.
4. Rev. 40: Suponemos la contratación de un mantenimiento evolutivo sobre la versión puesta en producción. Esta evolución debe ser paralela al desarrollo de la versión 2.x del producto en el Trunk, por lo que necesitará una Ramificación del Código implicando la creación de un Branch para la versión 1.x.
5. Rev. 50: Suponemos la detección de un bug en la versión en producción. Para resolverlo se deberá partir del código puesto en producción, por tanto se deberá recuperar el Tag de la versión 1.0.0. Para no perder la referencia a esta versión tras la realización de cambios se deberá hacer unaRamificación del código para resolver el bug en un Branch y realizar los cambios en dicha rama, con el objetivo de crear la versión 1.0.1.
6. Revs. 60–80: Trabajo en Equipo paralelo en el Trunk para la versión 2.x, en un Branch evolutivo para la versión 1.x y en un Branch correctiva para la versión 1.0.x.
7. Rev. 90: Cierre de Versión 1.0.1 del SW. Se crea el Tag de la versión 1.0.1.
8. Revs. 100-110: Debido a que tanto el desarrollo de la versión 1.x en el Branch como el de la versión 2.x en el Trunk han partido de la versión 1.0.0 que contenía el bug resuelto en la versión 1.0.1 será altamente probable la existencia del mismo error en estas ramas. Por tanto, se deberá hacer una Fusión de Cambios de las modificaciones realizadas desde la versión 1.0.0 a la versión 1.0.1 con el estado actual del Trunk y Branch.

Mercurial

 Es un [sistema de control de versiones](http://es.wikipedia.org/wiki/Sistema_de_control_de_versiones) [multiplataforma](http://es.wikipedia.org/wiki/Multiplataforma), para [desarrolladores de software](http://es.wikipedia.org/wiki/Desarrollador_de_software). Está implementado principalmente haciendo uso del [lenguaje de programación](http://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n) [Python](http://es.wikipedia.org/wiki/Python), pero incluye una implementación binaria de [diff](http://es.wikipedia.org/wiki/Diff" \o "Diff) escrita en [C](http://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n_C). Mercurial fue escrito originalmente para funcionar sobre Linux. Ha sido adaptado para [Windows](http://es.wikipedia.org/wiki/Microsoft_Windows), [Mac OS X](http://es.wikipedia.org/wiki/Mac_OS_X) y la mayoría de otros sistemas [tipo Unix](http://es.wikipedia.org/wiki/Unix). Mercurial es, sobre todo, un programa para la [línea de comandos](http://es.wikipedia.org/wiki/L%C3%ADnea_de_comandos). Todas las operaciones de Mercurial se invocan como opciones dadas a su programa motor, hg (cuyo nombre hace referencia al [símbolo químico](http://es.wikipedia.org/wiki/S%C3%ADmbolo_qu%C3%ADmico) del [mercurio](http://es.wikipedia.org/wiki/Mercurio_(elemento))).

Las principales metas de desarrollo de Mercurial incluyen un gran rendimiento y escalabilidad; desarrollo completamente distribuido, sin necesidad de un servidor; gestión robusta de archivos tanto de [texto](http://es.wikipedia.org/wiki/Archivo_de_texto) como [binarios](http://es.wikipedia.org/wiki/Archivo_binario); y capacidades avanzadas de ramificación e integración, todo ello manteniendo sencillez conceptual.[1](http://es.wikipedia.org/wiki/Mercurial#cite_note-1) Incluye una interfaz web integrada.

El creador y desarrollador principal de Mercurial es [Matt Mackall](http://es.wikipedia.org/w/index.php?title=Matt_Mackall&action=edit&redlink=1" \o "Matt Mackall (aún no redactado)). El [código fuente](http://es.wikipedia.org/wiki/C%C3%B3digo_fuente) se encuentra disponible bajo los términos de la licencia [GNU GPL](http://es.wikipedia.org/wiki/GNU_General_Public_License) versión 2, lo que clasifica a Mercurial como [software libre](http://es.wikipedia.org/wiki/Software_libre).

Características generales

* Mercurial usa resúmenes [SHA-1](http://es.wikipedia.org/wiki/Secure_Hash_Algorithm) para identificar revisiones.
* A diferencia de [git](http://es.wikipedia.org/wiki/Git" \o "Git) es muy fácil romper el repo, solo hace falta apretar ctrl+c en un commit y realizar un push para romper todos los repositorios.[[cita requerida](http://es.wikipedia.org/wiki/Wikipedia:Verificabilidad)]
* Para el acceso a repositorios mediante red, Mercurial usa un protocolo eficiente, basado en [HTTP](http://es.wikipedia.org/wiki/Hypertext_Transfer_Protocol), que persigue reducir el tamaño de los datos a transferir, así como la proliferación de peticiones y conexiones nuevas. Mercurial puede funcionar también sobre [ssh](http://es.wikipedia.org/wiki/Secure_Shell" \o "Secure Shell), siendo el protocolo muy similar al basado en HTTP.
* [Python](http://es.wikipedia.org/wiki/Python)
* [OpenOffice.org](http://es.wikipedia.org/wiki/OpenOffice.org)
* [Google Code](http://es.wikipedia.org/wiki/Google_Code) Project Hosting on Google Code permite crear repositorios mercurial (además de Subversion)
* [Aldrin](http://es.wikipedia.org/wiki/Aldrin)
* [Audacious](http://es.wikipedia.org/wiki/Audacious_Media_Player)
* [CubicWeb](http://es.wikipedia.org/wiki/CubicWeb)
* [Dovecot](http://es.wikipedia.org/wiki/Dovecot) [IMAP](http://es.wikipedia.org/wiki/IMAP) server[5](http://es.wikipedia.org/wiki/Mercurial#cite_note-5)
* [GNU Health](http://meta.wikimedia.org/wiki/w:en:GNU_Health)
* [GNU Octave](http://es.wikipedia.org/wiki/GNU_Octave)
* [Growl](http://es.wikipedia.org/wiki/Growl_(software))
* [MoinMoin](http://es.wikipedia.org/wiki/MoinMoin) wiki software
* [Mozilla](http://es.wikipedia.org/wiki/Mozilla_Firefox)[6](http://es.wikipedia.org/wiki/Mercurial#cite_note-6)
* [Netbeans](http://es.wikipedia.org/wiki/Netbeans)[7](http://es.wikipedia.org/wiki/Mercurial#cite_note-7)
* [OpenJDK](http://es.wikipedia.org/wiki/OpenJDK)[8](http://es.wikipedia.org/wiki/Mercurial#cite_note-8)
* [SAGE](http://es.wikipedia.org/wiki/Sage)
* [OpenSolaris](http://es.wikipedia.org/wiki/OpenSolaris) de Sun[9](http://es.wikipedia.org/wiki/Mercurial#cite_note-9)
* Los desarrolladores de [Xen](http://es.wikipedia.org/wiki/Xen)[10](http://es.wikipedia.org/wiki/Mercurial#cite_note-10)
* Software Opensource de [Oracle](http://es.wikipedia.org/wiki/Oracle) como [Btrfs](http://es.wikipedia.org/wiki/Btrfs)[11](http://es.wikipedia.org/wiki/Mercurial#cite_note-11)
* [Tuenti](http://es.wikipedia.org/wiki/Tuenti)
* Git

Git

Es un software de [control de versiones](http://es.wikipedia.org/wiki/Control_de_versiones) diseñado por [Linus Torvalds](http://es.wikipedia.org/wiki/Linus_Torvalds" \o "Linus Torvalds), pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando estas tienen un gran número de archivos de [código fuente](http://es.wikipedia.org/wiki/C%C3%B3digo_fuente). Al principio, Git se pensó como un motor de bajo nivel sobre el cual otros pudieran escribir la interfaz de usuario o [front end](http://es.wikipedia.org/wiki/Front-end_y_back-end" \o "Front-end y back-end) como [Cogito](http://es.wikipedia.org/wiki/Cogito_(software)) o [StGIT](http://es.wikipedia.org/w/index.php?title=StGIT&action=edit&redlink=1" \o "StGIT (aún no redactado)). [2](http://es.wikipedia.org/wiki/Git#cite_note-2)

Sin embargo, Git se ha convertido desde entonces en un sistema de control de versiones con funcionalidad plena.  Hay algunos proyectos de mucha relevancia que ya usan Git, en particular, el grupo de [programación](http://es.wikipedia.org/wiki/Programaci%C3%B3n) del [núcleo Linux](http://es.wikipedia.org/wiki/N%C3%BAcleo_Linux).

El [mantenimiento del software](http://es.wikipedia.org/wiki/Mantenimiento_de_Software) Git está actualmente supervisado por Junio Hamano, quien recibe contribuciones al código de alrededor de 280 programadores.

Características generales

* [Wiki](http://es.wikipedia.org/wiki/Wiki) para cada proyecto.
* [Página web](http://es.wikipedia.org/wiki/P%C3%A1gina_web) para cada proyecto.[1](http://es.wikipedia.org/wiki/GitHub#cite_note-1)
* Gráfico para ver cómo los desarrolladores trabajan en sus repositorios y [bifurcaciones](http://es.wikipedia.org/wiki/Bifurcaci%C3%B3n_(desarrollo_de_software)) del proyecto.
* Funcionalidades como si se tratase de una [red social](http://es.wikipedia.org/wiki/Red_social), como por ejemplo: seguidores.
* la ramificación de código sea con el objetivo de crear un proyecto nuevo a partir del código inicial, se deben considerar los Branches como ramas de desarrollo de vida limitada, es decir, tendrán un tiempo de vida tras el cual se deberá dejar de trabajar sobre ellos, bien por un Cierre de Versión o bien por la Fusión de Cambios a la rama de la que se hizo la ramificación.

Comandos básicos

* git fetch: Descarga los cambios realizados en el repositorio remoto.
* git merge: <nombre\_rama>:
* “nombre\_rama”: Impacta en la rama en la que te encuentras parado, los cambios realizados en la rama
* git pull: Unifica los comandos fetch y merge en un único comando.
* git commit -am “<mensaje>”: Confirma los cambios realizados. El “mensaje” generalmente se usa para asociar al commit una breve descripción de los cambios realizados.
* git push origin <nombre\_rama>: Sube la rama “nombre\_rama” al servidor remoto.
* git status: Muestra el estado actual de la rama, como los cambios que hay sin commitear.
* git add <nombre\_archivo>: Comienza a trackear el archivo “nombre\_archivo”.
* git checkout -b <nombre\_rama\_nueva>: Crea una rama, a partir de la que te encuentres parado, con el nombre
* “nombre\_rama\_nueva”, y luego salta sobre la rama nueva, por lo que quedas parado en ésta última.
* git checkout -t origin <nombre\_rama>:
* git branch: Si existe una rama remota de nombre “nombre\_rama”, al ejecutar este comando, te crea una rama local con el nombre “nombre\_rama” para trackear la rama remota con el mismo nombre.
* git branch -a: Lista todas las ramas locales.
* git branch -d <nombre\_rama>: Lista todas las ramas locales y remotas.
* git push origin :<nombre\_rama>: Elimina la rama local con el nombre “nombre\_rama”.