Bisoños Usuarios de GNU/Linux de Mallorca y Alrededores | Bergantells Usuaris de GNU/Linux de Mallorca i Afegitons

Darcs: control de versiones redux (22439 lectures)

Per **Carmen**, ***Carme*** (http://carme.balearweb.net)

Creado el 25/08/2006 07:41 modificado el 25/08/2006 07:41

Traducción del artículo[1] original de Pau Rul·lan[2]:

> *Darcs[3] es un sistema de control de versiones distribuido[4]. Este concepto juntamente de un desarrollo teórico denominado teoria de parches hacen de él un sistema a tener en cuenta.*

Presentación

Características de *Darcs*

Si quisiéramos enumerar las características de *Darcs* podríamos decir:

- Posibilidad de hacer *commits* locales (sin connexión)
- Se pueden deshacer *commits* concretos (el que se conoce como *cherry picking)*
- Cada repositorio es una rama por sí misma
- Independencia de un servidor central
- Simetria en los repositorios (todos siguen los mismos esquemas)
- Posibilidad de renombrar ficheros manteniendo el historial
- Podemos definir parámetros a nivel de usuarios o de repositorios
- Simplicidad infinita a la hora de poner en marcha un repositorio
- Suficientemente documentado (y sinó siempre teneis el código fuente)
- Varios métodos de acceso: local, ssh, http y ftp
- Soporte para ficheros binarios

El mayor incoveniente de la implementación actual de *Darcs* es el uso de memoria. Este es el punto donde se están poniendo todos los esfuerzos de desarrollo pero la situación actual no es satisfactoria por necesidades a gran escala (estoy hablando de repositorios de más de doscientos cincuenta megas en ficheros). Cuando hacemos el *checkout* inicial todo el repositorio subido ha de estar en memoria y eso es simplemente un gran problema. La memoria física tampoco facilita mucho: hemos de tener una copia de todo el repositorio y de todo el historial. Es decir, cuando hacemos una copia del repositorio no podemos bajarnos el *head* sino que nos lo hemos de comer todo. La consecuencia directa de esto es que proyectos grandes como serian el kernel de linux, los grandes escritorios o las X.org no se propondrán migrar a esta herramienta. Hay que decir que existe *git*, el programa que actualmente usan los desarrolladores del kernel, que aunque no es exactamente igual que darcs las directrices generales sí que son muy parecidas.

Nuevo concepto: control de versiones descentralizado

Despues de la hegemonia del CVS se creía que había poco que hacer con el tema de control de versiones. El conocidísimo Subversion, reemplazo lógico del CVS, simplemente reimplementaba el concepto de los vcs clásicos. Pero no siempre un control de versiones centralizado es la respuesta a nuestros problemas. Aquí es donde entran los controles de versiones descentralizados. Estos sistemas no solo representan una importante innovación al mundo de la ingenieria del software sino piden un cambio de *chip* y una manera de hacer las cosas un poco diferente a lo que

estamos acostumbrados.

La pieza clave de la descentralización del control de versiones es que no existe un único repositorio sino que **todos** los repositorios están al mismo nivel. Con esto quiere decir que cada repositorio, cada pull, es una rama distinta. De esta manera nos aseguramos una gran independencia entre desarrolladores, dando muchas facilidades no solo al desarrollador solitario y esporádico sino también al pequeño guerrero de code-monkeys. Hay que recordar que el hecho que sea descentralizado no nos limita a seguir los métodos y las técnicas de los sistemas centralizados como Subversion. Además, con lo que se conoce como *teoría de parches* podemos asegurar que en cualquier momento podemos meter nuestro trabajo en la rama de un repositorio que consideremos central.

Pondremos algunos ejemplos donde la descentralización nos puede ser útil.

- Queremos hacer un *commit* y no tenemos acceso (por falta de conexión, por ejemplo). A mi me gusta hacer un nuevo *record* (sinónimo de *commit* en la jerga de *Darcs*) cada vez que acabo de implementar una nueva funcionalidad que lleva trabajo o cuando he corregido un error grave.
- Acabamos de publicar una versión estable de nuestro programa y queremos dejar tranquilo el repositorio central. Como que cada copia es un repositorio por sí mismo no nos hemos de preocupar de nada.
- Estamos sin conexión y un nuevo recienllegado quiere colaborar. Él puede hacer un *pull* (sinónimo de *checkout* en *Darcs*) de *mi* rama y no de la del repositorio central. Esta situación se podría ver como una extensión de los casos anteriores.
- Tenemos un proyecto de software libre donde nosotros somos los principales desarrolladores pero queremos aceptar colaboraciones externas. Podemos usar un repositorio de *Darcs* de donde los usuarios potenciales puedan coger el código y enviarnos con mucha facilidad los cambios. Nos ahorramos mucho trabajo de gestión de permisos.
- Un sistema distribuido puede significar un acelerador para el encuentro y arreglo de fallos. Evidentemente se puede complementar con alguna herramienta tipo bugzila o mantis. La gracia es que al tener la estructura completa en la punta de los dedos y que con dos órdenes podemos enviar al administrador del repositorio los cambios, nos ahorraremos muchos pasos a la hora de corregir código. Y ya se sabe que un desarrollador contento es un desarrollador productivo.

Con todo recordar que la descentralización del repositorio es complementaria al modelo centralizado. Todo depende de nuestras prácticas y cómo lo usemos pero poder seguir con la rutina clásica si no nos sabemos acostumbrar al *Darcs*.

La pieza clave: la teoria de parches

Del inglés *patch theory*, la *teoria de parches* es todo un conjunto de pruebas y teoremas que permite la puesta en práctica de los sistemas de control de versiones descentralizados. Los tres conceptos importantes son el de parche, árbol y contexto. El **árbol** son el conjunto de ficheros y un **parche** es un cambio en el árbol. Esto lleva a que todo es un parche: añadir un fichero, borrarlo, renombrarlo, cambiarlo un poco o dejarlo sin contenido. Quedarse con esto: todo es un parche.
El **contexto** es el que ayuda la definición de la **representación** de un parche. Si la definición es lo que hace único y distinto a cada parche el contexto es el que determina dónde se ha de aplicar. Como que puede resultar muy complejo decidir donde comienza un árbol lo que se hace es que el estado inicial del repositorio sea un árbol vacío. Por eso hasta añadir ficheros es un parche.
El tema es muy interesante pero tampoco quiero entrar en detalles. Si quereis podeis seguir leyendo sobre ello en el mismo manual. A pesar de todo me gustaría comentar dos propiedades que facilitarán mucho la vida al administrador.

> La inversa de un parche es el parche más sencillo tal que la composición del original y su inverso no produce cambios en un árbol.

Como teorema añadimos que

> la inversa de la composición es la composición de los inversos.

Para hacer terriblement sencilla la resoución de conflictos tenemos otro teorema que, a groso modo, explica que los *parches son commutativos*. De esta manera nos evitamos el preocuparnos del orden en que aplicamos los parches e, incluso algunas veces, de aplicar todo el historial. Esto significa que de unos cambios a, b y c podríamos aplicar solamente a y c y no tendríamos ningún conflicto.

Ejemplo de uso

La manera más clara de aprender a usar un sistema de control de versiones es probarlo. Por este motivo he preparado y dejado un repositorio de *Darcs* donce todos vosotros podeis jugar. Yo también he ido haciendo cosas y dejo aquí algunas de las muestras.

Resumen de órdenes

Sin ser un substituto del manual este trozo podría ser una rápida referencia. Aquí teneis una para gente que viene de svn[5].

- help: la utilísima ayuda en línea.
- add: imprescindible si queremos añadir material.
- remove: desvincula del árbol sin borrar los ficheros. Si borrásemos el fichero directamente también desaparecería del árbol y quedaria constancia del borrado.
- mv: mueve, de verdad y no como hacen svn o cvs, ficheros.
- (un)revert: deshace o rehace los cambios en el árbol. Recordad que es el revert el que teneis que ejecutar si habeis borrado accidentalmente ficheros.
- (un)record: deja los cambios a un parche o aplica el invers de algún parche (es decir, deshace los cambios del parche ya aplicado).
- whatsnew: muestra las diferencias entre la situación actual y la última actualización.
- resolv: marca la resolución de conflictos.
- tag: deja una marca al historial del árbol. En realidad es un record sin modificaciones en los elementos.
- changes: muestra el historial de cambios.
- dist: hace un tar.gz del trunk del árbol.
- send: envia los parches generados por el record al mantenedor del repositorio.
- apply: aplicamos algún parche que nos hayan enviado.
- pull: actualiza el repositorio con base al pasado como parámetro. Esta orden deja marcado el último repositorio, así que no hará falta volver a escribirlo si no cambia la localización.
- push: sube los cambios grabados en el servidor padre.
- get: hacemos una rama del repositorio objetivo.
- put: complementario del get. Nos metemos en un repositorio y el parámetro es la localización del nuevo repositorio.

La sintaxis con los protocolos es sencilla pero se ha de saber:

- ssh: `darcs (pull|push) nomdusuari@maquina:cami/al/repo`
- http: `darcs (pull) http://maquina/cami/al/repo`
- ftp: `darcs (pull) ftp://nomdusuari:claudepass@maquina:cami/al/repo`
- ftp: `darcs (pull) ftp://nomdusuari@maquina:cami/al/repo (pedira contraseña)`
- correu: `darcs (send|apply)`

Este mismo artículo en un repo darcs

```
darcs initialize
echo "algun text per posar al nou fitxer" > primer-fitxer
darcs add -r .
darcs record  -la
```

Con esto ya tenemos nuestro repositorio inicializado y con el primer *commit*. Después de hacer el *record* se nos hará alguna pregunta como sería el correo electrónico del autor. Ahora podemos hacer el *pull* de una dirección http.

```
% mkdir apache-article-darcs
% cd apache-article-darcs
apache-article-darcs % darcs initialize
apache-article-darcs % darcs pull 'http://bulma.net/~paurullan/article-darcs/'

Sun Aug 20 20:25:43 CEST 2006  Pau Rul-lan Ferragut
```

```
* primer commit
  Shall I pull this patch? (1/1)  [ynWvpxqadjk], or ? for help: y
  Finished pulling and applying.

ssh-article-darcs % darcs push
Pushing to "bulma.net:/home/socis/paurullan/public_html/article-darcs"...

Sun Aug 20 20:46:30 CEST 2006  paurullan@bulma.net
  * afegit directori d'imatges
Shall I push this patch? (1/1)  [ynWvpxqadjk], or ? for help: a
Finished applying...
```

Con un acceso por ssh tendríamos los permisos que tenemos a la máquina remota, así que si lo teneis configurado podemos escribir.

```
% mkdir ssh-article-darcs
% cd ssh-article-darcs
ssh-article-darcs % darcs initialize
ssh-article-darcs % darcs pull
bulma.net:/home/socis/paurullan/public_html/article-darcs/
Sun Aug 20 20:25:43 CEST 2006  Pau Rul-lan Ferragut
* primer commit
Shall I pull this patch? (1/7)  [ynWvpxqadjk], or ? for help: a
Finished pulling and applying.

ssh-article-darcs % touch quart-fitxer
ssh-article-darcs % darcs add quart-fitxer
ssh-article-darcs % darcs record -al quart-fitxer
Recording changes in "quart-fitxer":

Darcs needs to know what name (conventionally an email address) to use as the
patch author, e.g. 'Fred Bloggs '.  If you provide one
now it will be stored in the file '_darcs/prefs/author' and used as a default
in the future.  To change your preferred author address, simply delete or edit
this file.

What is your email address? paurullan@bulma.net
What is the patch name? afegit quart fitxer
Do you want to add a long comment? [yn]n
Finished recording patch 'afegit quart fitxer'

ssh-article-darcs % darcs push -a
Pushing to "bulma.net:/home/socis/paurullan/public_html/article-darcs/"...
Finished applying...
```

Ramas locales, parte de la magia de los repos descentralizados. El comportamiento es igual al anterior solo que estaremos vinculados al ssh-article-darcs en lugar de al de la web.

```
% darcs get ssh-article-darcs copia-ssh-darcs
darcs get ssh-article-darcs copia-ssh-darcs
cd copia-ssh-darcs
touch segon-fitxer
darcs add -r .
darcs record
darcs push
```

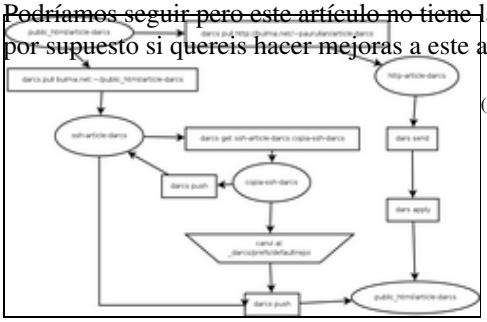
Cambiando los repositorios. Tened en cuenta que esto se habría de hacer tocando los ficheros de configuración y haciendo el push con el nuevo repositorio como parámetro.

```
copia-ssh-darcs % touch tercer-fitxer
copia-ssh-darcs % darcs add tercer-fitxer
copia-ssh-darcs % darcs record -al
What is the patch name? afegit tercer fitxer
Do you want to add a long comment? [yn]n
Finished recording patch 'afegit tercer fitxer'

copia-ssh-darcs % darcs push
```

```
Pushing to "/home/paurullan/experiment/ssh-article-darcs"...

Sun Aug 20 21:21:55 CEST 2006  paurullan@bulma.net
  * afegit tercer fitxer
Shall I push this patch? (1/1)  [ynWvpxqadjk], or ? for help: a
Finished applying...

(ara podriem fer: copia-ssh-darcs % darcs push
bulma.net:/home/socis/paurullan/public_html/article-darcs)
copia-ssh-darcs % echo "bulma.net:/home/socis/paurullan/public_html/article-darcs" >
 _darcs/prefs/defaultrepo
copia-ssh-darcs % darcs push
Pushing to "bulma.net:/home/socis/paurullan/public_html/article-darcs"...

Sun Aug 20 21:02:30 CEST 2006  paurullan@bulma.net
  * segon fitxer
Shall I push this patch? (1/2)  [ynWvpxqadjk], or ? for help: a
Finished applying...
```

Este sería un esquema de lo que hemos hecho con los repositorios. Fijaos como los círculos son repositorios por sí mismos. Las cajas son una pista de las órdenes que hemos ejecutado durante el ejemplo. Recordad que el cambio de objetivo al repositorio lo hemos hecho a mano y que sería lo mismo que hacer un `push http://bulma.net/~paurullan/article-darcs/`. También es importante notar que si queremos volver a actualizar nuestra copia-ssh-darcs tendremos que hacer un `pull` y no un `get`.

Podríamos seguir pero este artículo no tiene la intención de ser un manual para conocer los repositorios. Igualmente y por supuesto si quereis hacer mejoras a este artículo ya sabeis que teneis un árbol *Darcs solo para él* ^_^


(6)

Podeis hacer clic sobre la imagen para verlo en grande.

Referencias

Un par de páginas que os pueden ser útiles.

- Tabla comparativa de sistemas de control de versiones(4)
- Página oficial del programa(3)
- Wiki del programa(7)
- Interesante artículo en LinuxWeeklyNews(8)

Lista de enlaces de este artículo:

1. http://bulma.net/body.phtml?nIdNoticia=2333
2. http://bloc.balearweb.net/paurullan
3. http://abridgegame.org/darcs/
4. http://zooko.com/revision_control_quick_ref.html
5. http://darcs.net/DarcsWiki/MigratingFromSubversion
6. http://bulma.net/%7Epaurullan/article-darcs/imatges/diagrama-repositoris.png
7. http://darcs.net/DarcsWiki
8. http://lwn.net/Articles/109719/

E-mail del autor: cemaneiro _ARROBA_ gmail.com
Podrás encontrar este artículo e información adicional en: http://bulma.net/body.phtml?nIdNoticia=2335