Sistemi di Packaging

Emanuele Zamprogno - emanuele.zamprogno@fsugpadova.org

25 marzo 2009

Tabella dei contenuti

I pacchetti sono:

- Piccoli archivi contenenti dei file binari, ordinati in cartelle specifiche
- L'archivio segue una logica definita dal tipo di dato che contiene e dalla logica impostagni dal package manager

[1] Un piccolo file di testo

I pacchetti sono:

- Piccoli archivi contenenti dei file binari, ordinati in cartelle specifiche
- L'archivio segue una logica definita dal tipo di dato che contiene e dalla logica impostagni dal package manager
- Talvolta può non essere un archivio ma uno **Script**[1], contenente le istruzioni da far eseguire al package manager

[1] Un piccolo file di testo

I pacchetti sono:

- Piccoli archivi contenenti dei file binari, ordinati in cartelle specifiche
- L'archivio segue una logica definita dal tipo di dato che contiene e dalla logica impostagni dal package manager
- Talvolta può non essere un archivio ma uno **Script**[1], contenente le istruzioni da far eseguire al package manager

[1] Un piccolo file di testo

I pacchetti sono:

- Piccoli archivi contenenti dei file binari, ordinati in cartelle specifiche
- L'archivio segue una logica definita dal tipo di dato che contiene e dalla logica impostagni dal package manager
- Talvolta può non essere un archivio ma uno **Script**[1], contenente le istruzioni da far eseguire al package manager

[1] Un piccolo file di testo

lavorando da terminale

1. Con Debian e derivate: **apt-get install foo..**
2. Con Fedora e derivate: **yum install foo..**

lavorando da terminale

1. Con Debian e derivate: **apt-get install foo..**
2. Con Fedora e derivate: **yum install foo..**
3. Con Gentoo: **emerge foo..**

lavorando da terminale

1. Con Debian e derivate: **apt-get install foo..**
2. Con Fedora e derivate: **yum install foo..**
3. Con Gentoo: **emerge foo..**

É altrimenti possibile lavorare con tool grafici che non sono altro che interfacce per i tool da CLI

lavorando da terminale

1. Con Debian e derivate: **apt-get install foo..**
2. Con Fedora e derivate: **yum install foo..**
3. Con Gentoo: **emerge foo..**

É altrimenti possibile lavorare con tool grafici che non sono altro che interfacce per i tool da CLI

Installare programmi SENZA un Package Manager

Pensate come s'installa un programma su un qualsiasi Windows[2] :), e pensate alle **DLL** che fanno a pugni fra loro

[2] si proprio su tutti, anche Vista

Un solo sorgente per compilarli tutti!

- Normalmente, nelle community FOSS, un programma o un driver viene distribuito da chi lo scrive in formato sorgente quindi per usarlo é l'utente che lo deve compilare.
- nelle distro questa impellenza viene assunta dai manutentori della distribuzione stessa, che si accollano anche il problema del **Dependency Hell**[3]

[3] http://en.wikipedia.org/wiki/Dependency_hell

Un solo sorgente per compilarli tutti!

- Normalmente, nelle community FOSS, un programma o un driver viene distribuito da chi lo scrive in formato sorgente quindi per usarlo é l'utente che lo deve compilare.
- nelle distro questa impellenza viene assunta dai manutentori della distribuzione stessa, che si accollano anche il problema del **Dependency Hell**[3]
- Si ha quindi un albero di pacchetti già compilati e con i corretti rapporti delle dipendenze, compilati ad hoc per l'architettura di destinazione:
  Mettiamo questo albero on line e avremo un Repository!!

[3] http://en.wikipedia.org/wiki/Dependency_hell

Un solo sorgente per compilarli tutti!

- Normalmente, nelle community FOSS, un programma o un driver viene distribuito da chi lo scrive in formato sorgente quindi per usarlo é l'utente che lo deve compilare.
- nelle distro questa impellenza viene assunta dai manutentori della distribuzione stessa, che si accollano anche il problema del **Dependency Hell**[3]
- Si ha quindi un albero di pacchetti già compilati e con i corretti rapporti delle dipendenze, compilati ad hoc per l'architettura di destinazione:
  Mettiamo questo albero on line e avremo un Repository!!

[3] http://en.wikipedia.org/wiki/Dependency_hell

Un solo sorgente per compilarli tutti!

- Normalmente, nelle community FOSS, un programma o un driver viene distribuito da chi lo scrive in formato sorgente quindi per usarlo é l'utente che lo deve compilare.
- nelle distro questa impellenza viene assunta dai manutentori della distribuzione stessa, che si accollano anche il problema del **Dependency Hell**[3]
- Si ha quindi un albero di pacchetti già compilati e con i corretti rapporti delle dipendenze, compilati ad hoc per l'architettura di destinazione:
  Mettiamo questo albero on line e avremo un Repository!!

[3] http://en.wikipedia.org/wiki/Dependency_hell

Ἐν αρχη ἠν ὁ Sorgente!

- Come sopra detto un **Repostory è un insieme ordinato di pacchetti contenente un set di meta-informazioni dedicate al mantenerlo ordinato e aggiornabile.**
- Queste informazioni posso essere inserite in un set di file dedicati, come avviene nel portage di Gentoo o in OpenEmbedded, oppure all'interno di ogni singolo pacchetto, o anche in una struttura di DB relazionale[4]

[4] come nel caso del BerkleyDB dei DPKG

Ἐν αρχη ἠν ὁ Sorgente!

- Come sopra detto un **Repostory è un insieme ordinato di pacchetti contenente un set di meta-informazioni dedicate al mantenerlo ordinato e aggiornabile.**
- Queste informazioni posso essere inserite in un set di file dedicati, come avviene nel portage di Gentoo o in OpenEmbedded, oppure all'interno di ogni singolo pacchetto, o anche in una struttura di DB relazionale[4]
- Importanti differenze sono invece presenti in tutta la famiglia dei ***BSD** e in Gentoo Linux, usando il meccanismo di **Ports**

[4] come nel caso del BerkleyDB dei DPKG

Ἐν αρχη ἠν ὁ Sorgente!

- Come sopra detto un **Repostory è un insieme ordinato di pacchetti contenente un set di meta-informazioni dedicate al mantenerlo ordinato e aggiornabile.**
- Queste informazioni posso essere inserite in un set di file dedicati, come avviene nel portage di Gentoo o in OpenEmbedded, oppure all'interno di ogni singolo pacchetto, o anche in una struttura di DB relazionale[4]
- Importanti differenze sono invece presenti in tutta la famiglia dei ***BSD** e in Gentoo Linux, usando il meccanismo di **Ports**

[4] come nel caso del BerkleyDB dei DPKG

Ἐν αρχη ἠν ὁ Sorgente!

- Come sopra detto un **Repostory è un insieme ordinato di pacchetti contenente un set di meta-informazioni dedicate al mantenerlo ordinato e aggiornabile.**
- Queste informazioni posso essere inserite in un set di file dedicati, come avviene nel portage di Gentoo o in OpenEmbedded, oppure all'interno di ogni singolo pacchetto, o anche in una struttura di DB relazionale[4]
- Importanti differenze sono invece presenti in tutta la famiglia dei ***BSD** e in Gentoo Linux, usando il meccanismo di **Ports**

[4] come nel caso del BerkleyDB dei DPKG

RPM, DEB, ARCH, una cibalgina per favore..

1. Per motivi storici, e non tecnici[5], esistono diversi sistemi di packaging. **Fra questi RPM è sicuramente il primo ad essere stato presentato, ma ha dovuto penare per avere un supporto accettabile al concetto di repository.**
2. Il secondo in ordine di tempo è stato il Deb, che grazie soprattuto al sistema APT che ha integrato da subito il concetto di repository nell'ottica dei Deb, traendone beneficio non piccolo

[5] tranne rarissime eccezioni

RPM, DEB, ARCH, una cibalgina per favore..

1. Per motivi storici, e non tecnici[5], esistono diversi sistemi di packaging. **Fra questi RPM è sicuramente il primo ad essere stato presentato, ma ha dovuto penare per avere un supporto accettabile al concetto di repository.**
2. Il secondo in ordine di tempo è stato il Deb, che grazie soprattuto al sistema APT che ha integrato da subito il concetto di repository nell'ottica dei Deb, traendone beneficio non piccolo
3. Ricordiamo sempre però che le mainline per i sistemi di packaging sono a tutt'oggi due:

[5] tranne rarissime eccezioni

RPM, DEB, ARCH, una cibalgina per favore..

1. Per motivi storici, e non tecnici[5], esistono diversi sistemi di packaging. **Fra questi RPM è sicuramente il primo ad essere stato presentato, ma ha dovuto penare per avere un supporto accettabile al concetto di repository.**
2. Il secondo in ordine di tempo è stato il Deb, che grazie soprattuto al sistema APT che ha integrato da subito il concetto di repository nell'ottica dei Deb, traendone beneficio non piccolo
3. Ricordiamo sempre però che le mainline per i sistemi di packaging sono a tutt'oggi due:
   1. la Red Hat Based
   2. la Debian Based

[5] tranne rarissime eccezioni

RPM, DEB, ARCH, una cibalgina per favore..

1. Per motivi storici, e non tecnici[5], esistono diversi sistemi di packaging. **Fra questi RPM è sicuramente il primo ad essere stato presentato, ma ha dovuto penare per avere un supporto accettabile al concetto di repository.**
2. Il secondo in ordine di tempo è stato il Deb, che grazie soprattuto al sistema APT che ha integrato da subito il concetto di repository nell'ottica dei Deb, traendone beneficio non piccolo
3. Ricordiamo sempre però che le mainline per i sistemi di packaging sono a tutt'oggi due:
   1. la Red Hat Based
   2. la Debian Based

[5] tranne rarissime eccezioni

Non solo le distro hanno i pacchetti

Ricordiamo che molti progetti a livello applicazione hanno un set di pacchetti organizzati e di repository dedicati come avviene per:

- il sistema **R**
- il linguaggio Perl
- il sistema LaTeX 2ε
- il Linguaggio Python
- il Framework .NET di MS

Non solo le distro hanno i pacchetti

Ricordiamo che molti progetti a livello applicazione hanno un set di pacchetti organizzati e di repository dedicati come avviene per:

- il sistema **R**
- il linguaggio Perl
- il sistema LaTeX 2_ε
- il Linguaggio Python
- il Framework .NET di MS

le cinque W di **/opt**

- Spesso i sistemi di packaging non nativi usano la cartella /opt che sta per **Optional** invenzione di SUN ancora ai tempi di SunOS
- Viene usata dentro MacOS X, per il sistema di **MacPorts**[6]

[6] Prima conosciuto come Darwin ports

le cinque W di **/opt**

- Spesso i sistemi di packaging non nativi usano la cartella /opt che sta per **Optional** invenzione di SUN ancora ai tempi di SunOS
- Viene usata dentro MacOS X, per il sistema di **MacPorts**[6]
- Ultimamente anche nei sistemi Win32 ci sono iniziative per inserire delle logiche più sane per gestire gli applicativi presenti nel sistema.

[6] Prima conosciuto come Darwin ports

le cinque W di **/opt**

- Spesso i sistemi di packaging non nativi usano la cartella /opt che sta per **Optional** invenzione di SUN ancora ai tempi di SunOS
- Viene usata dentro MacOS X, per il sistema di **MacPorts**[6]
- Ultimamente anche nei sistemi Win32 ci sono iniziative per inserire delle logiche più sane per gestire gli applicativi presenti nel sistema.
- Ultimo ma non ultimo l'ideona di **CygWin** che negli anni '90 fece furore, ed era una specie di opt nei sistemi win32.

[6] Prima conosciuto come Darwin ports

le cinque W di **/opt**

- Spesso i sistemi di packaging non nativi usano la cartella /opt che sta per **Optional** invenzione di SUN ancora ai tempi di SunOS
- Viene usata dentro MacOS X, per il sistema di **MacPorts**[6]
- Ultimamente anche nei sistemi Win32 ci sono iniziative per inserire delle logiche più sane per gestire gli applicativi presenti nel sistema.
- Ultimo ma non ultimo l'ideona di **CygWin** che negli anni '90 fece furore, ed era una specie di opt nei sistemi win32.

[6] Prima conosciuto come Darwin ports

le cinque W di **/opt**

- Spesso i sistemi di packaging non nativi usano la cartella /opt che sta per **Optional** invenzione di SUN ancora ai tempi di SunOS
- Viene usata dentro MacOS X, per il sistema di **MacPorts**[6]
- Ultimamente anche nei sistemi Win32 ci sono iniziative per inserire delle logiche più sane per gestire gli applicativi presenti nel sistema.
- Ultimo ma non ultimo l'ideona di **CygWin** che negli anni '90 fece furore, ed era una specie di opt nei sistemi win32.

[6] Prima conosciuto come Darwin ports

Come diceva Ian Murdock

["Package Management is] *the single biggest advancement Linux has brought to the industry", that it blurs the boundaries between operating system and applications, and that it makes it* **"easier to push new innovations [...] into the marketplace and [...] evolve the OS".**

A voi la parola!

E grazie per la pazienza nell'ascolto!