Pacchettizzazione e distribuzione del software

Gabriele D'Angelo
<gda@cs.unibo.it>
http://www.cs.unibo.it/~gdangelo

Università degli Studi di Bologna
Dipartimento di Scienze dell'Informazione

Aprile, 2005

Scaletta della lezione

- Definizione del problema
- Rilasciare solamente i sorgenti
- Rilasciare anche i binari
- Packaging system
- Alcuni esempi poco tradizionali
- Open Source & Microsoft Windows?
- Breve bibliografia

Definizione del problema

- Abbiamo scritto un nuovo bellissimo software, la licenza ovviamente è Open Source ... come lo distribuiamo?

- Alcune possibilità:
  - rilasciamo solo i sorgenti, chi vuole usarlo è costretto a ricompilarlo sulla propria architettura
  - rilasciamo anche i binari:
    - pacchetti? Per quali distribuzioni?
    - architettura? Solo i386?
    - sistema operativo? Non esiste sono GNU/Linux!

“Release early, release often” ... quanto ci “costa”?

Rilasciare solamente sorgenti

La soluzione banale (ed infatti molto diffusa) è quella di rilasciare solamente un .tgz che contiene tutti i sorgenti del progetto

Chi vuole utilizzare il nostro software deve:

- avere un ambiente di sviluppo installato (adeguato)
- eventualmente effettuare un porting di alcune parti
- procedere alla compilazione e all'installazione

Le richieste sono decisamente troppe, molti saranno scoraggiato dal provare il nostro software.

Meno persone provano il software, minori sono le possibilità che qualcuno collabori allo sviluppo / porting

Rilasciare binari

Visto che rilasciare solamente i sorgenti non sembra un approccio promettente, fissato sistema operativo e architettura hardware potremmo rilasciare i binari.

In quale formato?

- .tgz che contiene i binari
- pacchetto di una o più distribuzioni

Costruire un .tgz è comodo e veloce ma:

- non tiene conto delle dipendenze quindi potrebbe non funzionare
- rischia di “sporcare” il sistema e renderlo incoerente

Packaging system

Cerchiamo di capire bene cos'è un pacchetto con un esempio:

http://packages.debian.org/unstable/gnome/evolution

- Dipendenze, raccomandati, suggeriti
- Il pacchetto è presente per varie architetture
- Proviamo a scaricare un pacchetto e a "navigare" al suo interno
- Troviamo il software vero e proprio ed un insieme script / meta informazioni
- I pacchetti delle varie distribuzioni sono estremamente comodi perchè ci permettono di gestire: le dipendenze, effettuare pre-patch dei sorgenti, seguire policy, effettuare dei rebuild con relativamente poco sforzo

Packaging system

Sfortunatamente non è tutto così semplice:

- non esiste uno standard (approvato e seguito da tutti) per fissare il nome e il contenuto dei pacchetti, le distribuzioni utilizzano nomi diversi e dividono il software in pacchetti diversi
- non solo … cambiando distribuzione molto spesso cambia anche l' "aspetto del sistema", i file di configurazione sono in posti diversi ecc.

- Non possiamo mischiare con successo dep, tgz, rpm… nel caso degli rpm la situazione è ancora peggiore. Gli rpm sono teoricamente multi-distribuzione, nella realtà questo non succede

- Costruire un pacchetto è molto costoso in termini di tempo

Packaging system

Una soluzione brutale è costituita dal software “alien”. Tenta (a volte con successo, altre volte molto meno) di effettuare traduzioni tra un formato e l’altro

Come visto le distribuzioni sono intrinsecamente troppo diverse perchè uno strumenti di questo genere possa avere sempre successo

Quali alternative abbiamo a questa situazione?

- con più collaborazione tra le varie distribuzioni (auspicabile, realizzabile?)
- cercando di cambiare paradigma (seguono alcune idee)

Autopackage

- Autopackage http://www.autopackage.org

  L'idea è quella di costruire un file (.package) che contiene tutti gli altri file necessari all'esecuzione del software che intendiamo distribuire. Tutto questo in maniera del tutto neutrale dalla distribuzione che ci ospita. Anche in questo caso nella pratica si tratta di un tarball ed uno script

Estendendo il concetto, lo script potrebbe verificare il sistema ospite e scaricare automaticamente altro software necessario per l'esecuzione ma attualmente assente.

Autopackage è indipendente dalla distribuzione e definisce un proprio ambiente di installazione / gestione / configurazione

Zero-Install

- Zero-Install http://zero-install.sourceforge.net

  La nuova incarnazione di un'idea vecchia: il software non è installato ma viene scaricato su necessità

  - Non si installa nulla, si fa solo cache di quanto scaricato
  - Visto che non è necessario installare, non ci sono problemi di dipendenze in locale
  - I vecchi problemi di sicurezza sono sostituiti da nuovi, diversi
  - Non è necessario avere privilegi particolari per utilizzare un nuovo software

Esecuzione browser-based

Se pensate che la proposta precedente sia troppo “lontana” dall’informatica attuale forse è bene riflettere un attimo sul ruolo dei browser

Uno dei trend riconosciuti ed a maggiore crescita è la volontà di migrare sempre più applicazioni sul web (abbiamo visto che anche subversion può essere accessibile da http, https)

Un esempio su tutti: la webmail

Mozilla Firefox può essere un’ottima base per la costruzione di applicazioni cross-platform! Estensioni, XUL ecc.

Open Source & Microsoft Windows

Ha senso sviluppare software Open Source per Microsoft Windows?

Se la riposta è sì allora abbiamo la necessità di un installer anche per questa piattaforma. Paradossalmente in questo caso tutto appare molto più semplice a causa della forte omogeneità

Non ci resta che cercare un installer Open Source e provarlo

Ad esempio il Windows Installer XML (WiX) toolset

http://sourceforge.net/project/wix/

... gentilmente offerto da?

Note e bibliografia

- Debian New Maintainers' Guide. http://www.debian.org/doc/maint-guide/
- Debian Binary Package Building HOWTO
  http://www.ibiblio.org/pub/Linux/docs/HOWTO/other-formats/html_single/Debian-Binary-Package-Building-HOWTO.html
- Mozilla Projects. http://www.mozilla.org/projects/