

Installazione di software su Linux, presente e futuro

Perché il software che serve non si trova per la propria versione di Linux? Perché installarlo non è semplice come avviene sui sistemi basati su Windows oppure Mac OS X? Computer e sistemi operativi sono utili solo se ci si possono far girare facilmente le *applicazioni*, cioè i programmi software scritti per svolgere una funzione specifica. Di conseguenza, uno dei fattori più importanti che ci

fanno scegliere una versione di Linux o un'altra è quanto sia facile *trovare e installare* i programmi di cui si ha bisogno.

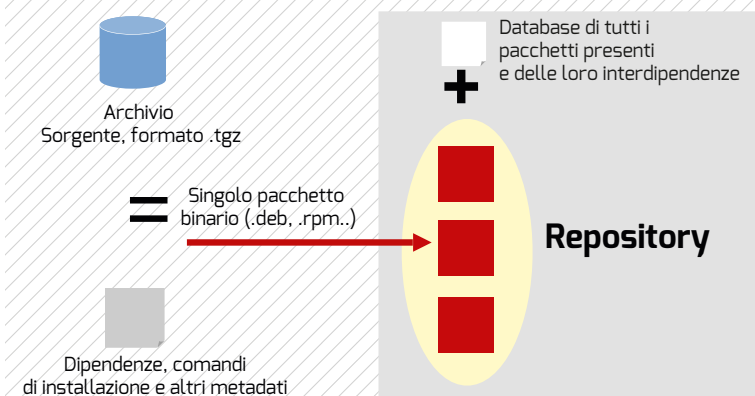
Da questo punto di vista Linux è in una situazione abbastanza paradossale. I suoi metodi di scoperta e installazione software sono ormai sufficientemente stabili e sofisticati da non creare problemi a tantissimi utenti, anche

Le tante distribuzioni e numerose versioni del sistema operativo rendono delicato e critico installare correttamente le applicazioni necessarie.

alle prime armi. Allo stesso tempo, però, hanno conseguenze tali per gli sviluppatori, da scoraggiare, per così dire, anche lo stesso creatore di Linux, Linus Torvalds.

Pochi lo sanno, ma questo eccellente programmatore è anche l'autore di Sub Surface (<http://subsurface-divelog.org>), un'applicazione multiplatforma per archiviazione e analisi dei dati di immersioni. La notizia è che Torvalds, mentre produce personalmente versioni installabili di Sub Surface per OS X e Windows, non lo fa per Linux, limitandosi a fornire il solo codice sorgente! Perché questo avviene? Scopriamo perché il problema non è così grave come lo racconta Torvalds (anche se non va ignorato), e infine una possibile soluzione che ha già generato parecchio interesse e polemiche.

COME È CAMBIATA NEL TEMPO LA PROCEDURA



L'evoluzione storica dell'installazione di software sotto Linux: dagli archivi compressi di codice da compilare e installare a mano alle repository online di pacchetti binari.

COME SI DISTRIBUISCE IL SOFTWARE PER LINUX

Consentiteci una sorta di nota: anche se a moltissimi utenti capita o capiterà di imbattersi nei sistemi qui descritti *solo* in Ubuntu o qualche altra distribuzione del genere, e se qui continueremo a scrivere "Linux" per comodità, non c'è nulla di specificamente Linux in quanto segue, anzi.

Linux in senso stretto è solo un kernel, non un sistema operativo completo, parleremo di standard e procedure utilizzate in qualsiasi ambiente simil-Unix e Open Source. Questo include tutte le distribuzioni Gnu/Linux vere e proprie, l'ambiente Hurd dello stesso progetto Gnu e tutta la famiglia di sistemi operativi BSD.

Premesso questo, potremmo dire che scoperta e installazione di software sotto Linux sono vittime del successo delle premesse fondamentali da cui sono nate! Sotto Linux queste operazioni seguono sempre pochi principi fondamentali, praticabili perché, trattandosi di codice aperto, non esistono barriere sostanziali alla sua condivisione.

La compattezza (un programma per Linux è spesso molto meno ingombrante della stessa versione per Windows) è dovuta proprio al riutilizzo dei componenti. Sotto Linux si installano separatamente, per riusarli e dividerli il più possibile, *tutti* i componenti comuni a due o più programmi, chiunque ne sia l'autore. Questo vale per qualsiasi tipo di libreria o altri componenti: icone, algoritmi di cifratura, sfondi per finestre, operatori matematici, dizionari, decoder audio/video, tutto è compilato e installato in modo indipendente.

Questo modo di lavorare offre enormi vantaggi dal punto di vista della manutenzione e della sicurezza. Se una libreria utilizzata da dieci programmi diversi dev'essere aggiornata, magari perché vulnerabile ad attacchi informatici, basta scaricarne una sola copia, una volta sola, per essere tranquilli. Gli sviluppatori possono anch'essi ignorare il problema di sicurezza del nostro esempio, proprio perché sanno che l'autore della libreria avrà sia la possibilità sia la responsabilità di

aggiornarla. Il lato "oscuro" di questa soluzione è che molte applicazioni sembrano piccole solo perché prima di utilizzarle bisogna avere già scaricato a parte altre decine o centinaia di Megabyte di librerie comuni. Questo è un problema più teorico che pratico, almeno per chi ha una buona

connessione Internet, e che si presenta solo quando si installa il primo di tutti i programmi che hanno bisogno di una determinata libreria.

La libertà di compilare i sorgenti permette infine di creare i file eseguibili più adatti alla combinazione di distribuzione, Cpu, memoria e chipset grafici su cui dovranno girare. Prodotti come Gentoo sfruttano questo principio al massimo.

IN PRINCIPIO C'ERANO GLI ARCHIVI

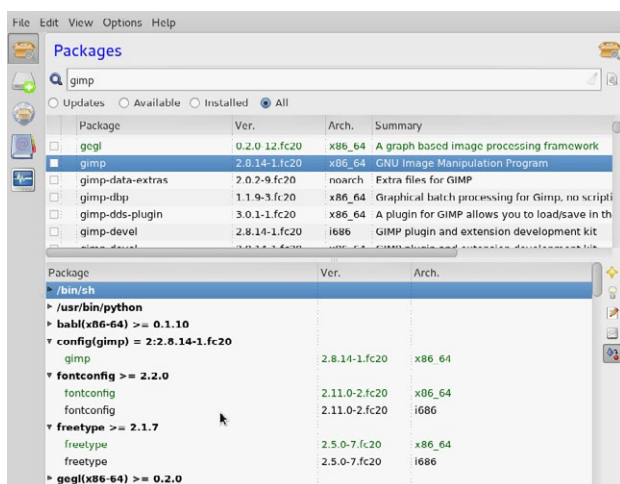
Il sistema originario di installazione software su Linux è ancora disponibile oggi, oltre a essere la norma su una distribuzione storica come Slackware: archivi compressi di codice sorgente, in formato .tgz, completi di tutte le istruzioni per compilarli. Un metodo allo stesso livello, dal punto di vista dell'integrazione con il resto del software, è quello applicato negli archivi con estensione .sh, che però opera in maniera completamente diversa. Per installare a partire da file .tgz occorre già avere un compilatore e saperlo

Torvalds scontento

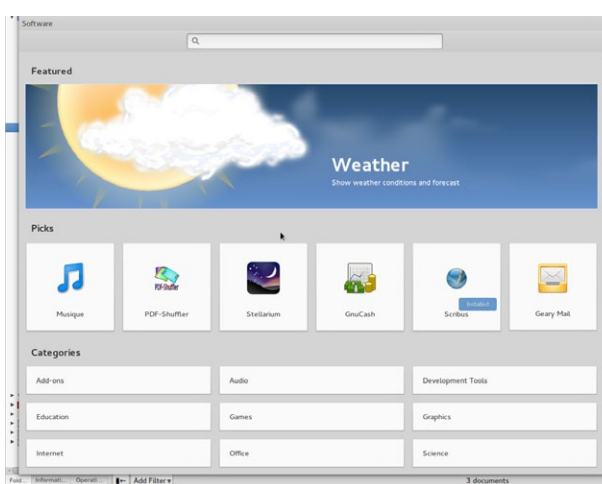
Anche il creatore di Linux riconosce la difficoltà delle procedure di installazione

COME SI INSTALLA SU WINDOWS

Ivari installer disponibili per Windows funzionano più o meno come i moderni gestori software per Linux con interfaccia grafica, utilizzando un formato file, con estensione .msi, simile a quelli descritti in questo articolo. All'interno di un file .msi si trovano i comandi per installare e disinstallare un programma, oltre al suo file eseguibile vero e proprio e a tutti quelli ausiliari, dalle icone agli sfondi, di cui ha bisogno per funzionare. La differenza primaria rispetto a Linux è che molto spesso, per i programmi Windows, il file eseguibile, o quelli "ausiliari" contengono anche copie complete di tutte le librerie che servono per girare. Questo è dovuto (in misura variabile da programma a programma) a ragioni commerciali o legali, alla volontà di risparmiare all'utente qualsiasi passo intermedio e al bisogno di minimizzare le richieste di supporto, cioè di non dover fare i conti con più possibili configurazioni software. È sostanzialmente per questo che un programma per Windows si può (quasi sempre) installare con pochi clic. Un'altra differenza rispetto a Linux, più visibile ma tutto sommato meno profonda e importante, è la cosiddetta "esperienza" che è possibile offrire all'utente durante l'installazione. Gli installer per Windows facilitano molto l'inserzione di messaggi, anche pubblicitari, e in genere la personalizzazione grafica delle varie schermate in cui occorre fare clic.



Sono finiti i tempi in cui i pacchetti da installare si dovevano scovare uno per uno. Ora i front-end come Synaptic, Apter o Yum-Extender, svolgono il compito con un solo clic.



Su distribuzioni come Fedora, nell'illustrazione, o Ubuntu, scoperta e installazione di programmi avvengono già con interfacce facili da usare quasi quanto quelle per smartphone.

usare, entrambe cose non più scontate fra gli utenti Linux di oggi: bisogna infatti generare da sé la versione eseguibile del programma, magari personalizzandola. I file con estensione .sh sono invece installatori di *software già compilato*.

Per servirsene basta lanciarli da riga di comando e rispondere, sempre nel terminale, a qualche domanda su dove collocare i file eseguibili e quelli di configurazione. Sia gli archivi .tgz sia quelli .sh (ma lo stesso discorso varrebbe per tante applicazioni multipiattaforma Java distribuite in file .jar) hanno lo stesso limite: è garantito che tutto andrà come dovrebbe andare solo se l'utente finale ha esattamente lo stesso insieme di software e le stesse configurazioni di sistema, dello sviluppatore che ha preparato e distribuito l'archivio.

Se questo non avviene, e accade spesso, le cose potrebbero ancora funzionare a patto di essere disposti a installare o configurare manualmente altri pacchetti (dopo aver scoperto da soli

quali sono e dove ottenerli, cosa non sempre facile), se non addirittura a modificare il codice sorgente.

POI SONO ARRIVATI I PACCHETTI

Già sul finire del secolo scorso il mondo Linux aveva superato i limiti che abbiamo appena citato con l'introduzione dei pacchetti binari, ovvero quei file con estensione .rpm per Red Hat, Fedora, CentOS e derivati, o .deb per Debian, Ubuntu e tutti i loro cloni (i file .ppa, ovvero i *Personal Package Archive* di Ubuntu, sono sempre .deb, ma forniti da singoli utenti). Questi formati di pacchetti hanno fatto passi da gigante rispetto al passato, in termini di semplicità d'uso e di evitare brutte sorprese e frustranti cacce online a informazioni e software mancanti. Ciò è dovuto innanzitutto al fatto che sia i file .deb che quelli .rpm contengono versioni di un certo programma compilate specificamente per una singola versione di una determinata



L'interfaccia Ubuntu Publish promette di semplificare notevolmente la preparazione di pacchetti binari nel nuovo formato Click per Ubuntu Touch.

CLICK, APPLICAZIONI CHIAVI IN MANO PER UBUNTU?



La distribuzione Ubuntu Touch, realizzata per smartphone, ha un nuovo formato di pacchetti di installazione.

C'è un sistema operativo che nel 2015 potrebbe arrivare nelle mani di moltissime persone e rivoluzionare anche il sistema attuale di creazione, distribuzione e installazione di software sotto Linux. Ubuntu Touch non è una variante di Android, ma una vera e propria distribuzione Linux fatta apposta per smartphone e tablet, che potrebbe essere disponibile per vari modelli di terminali mobili nel 2015. Le parti di Ubuntu Touch più rilevanti per il tema di questo mese sono un nuovo formato di pacchetti pensato soprattutto per terminali mobili, chiamato Click, e soprattutto l'App Store ufficiale di Ubuntu ad esso associato. Presi insieme, i pacchetti Click e lo store offriranno procedure di impacchettamento, distribuzione e installazione software molto più simili a quelle iPhone o Android che agli ambienti tradizionali Linux costituiti da repository e gestori di pacchetti come Synaptic. Come spiegheremo fra poco, le differenze sostanziali fra quei sistemi e Click potranno apprezzarle prima gli sviluppatori, e forse molto di più, degli utenti finali. I cambiamenti sono però tali da poter avere conseguenze importanti anche per tutti gli utenti Linux (Ubuntu 14.10, per esempio, è già compatibile con Click).

COME FUNZIONA CLICK?

Le specifiche del formato e delle procedure dell'App Store ufficiale descrivono come creare, distribuire e installare pacchetti in una maniera molto più simile a quella standard per Windows che alle pratiche Linux descritte nell'articolo principale. Un programma impacchettato con Click e pubblicato nel Click App Store, infatti, è per prima cosa *privo di dipendenze*, a parte quella ovvia dalla versione base della distribuzione per cui è stato creato. Invece di contare su librerie condivise, ogni pacchetto Click arriva con una *sua* copia di tutto quel che serve al programma. Anche per questo la procedura d'installazione, a differenza di quanto avviene con pacchetti .rpm o .deb, non può eseguire comandi o script che sarebbero potenzialmente



Il Click App Store di Ubuntu potrebbe rendere l'installazione di software su terminali Ubuntu Touch facile quanto quella su Android o iOS.

distribuzione. Queste informazioni, per evitare ambiguità, sono spesso presenti nel nome dei pacchetti stessi. In secondo luogo entrambi i formati permettono, per non dire impongono, un'installazione più automatizzata possibile. Ogni pacchetto, infatti, contiene in formati standard non solo i comandi per copiare i vari file eseguibili nelle cartelle giuste, ma anche quelli che servono per configurarli. Questo permette a software di gestione pacchetti, come rpm in un caso o dpkg nell'altro, di eseguire tutte le operazioni da soli, senza che l'utente si accorga di nulla.

In teoria, la categoria dei "comandi di configurazione" può contenere di tutto o quasi, inclusi generici script da eseguire con i privilegi da amministratore. Normalmente però si tratta di impostazione dei permessi di accesso, avvio automatico a ogni accensione

o a intervalli predefiniti (*cron job*) nel caso di server, oppure creazione di utenti speciali per svolgere attività particolari.

Ancora più importanti dei comandi sono i metadati: standardizzare la distribuzione di queste informazioni è vitale per la sanità mentale degli amministratori di sistema. Purtroppo, per quanto strano

possa sembrare oggi, arrivarci è stato un cammino lungo, tormentato e non ancora completamente concluso.

Alcuni metadati sempre presenti sembrano banali, anche se è difficile minimizzare l'importanza: firma digitale per evitare manomissioni, numero completo di versione, sito di provenienza e licenza. Seguono a ruota il nome del creatore del *pacchetto*, che quasi mai è lo sviluppatore originale del *programma*, e una breve descrizione di quest'ultimo per capire subito cosa fa.

Come iOS e Android

L'App Store di Ubuntu promette di rendere semplicissima la selezione e installazione dei programmi.

pericolosi. Sempre per ragioni di sicurezza il Click Apple Store garantisce che tutti i programmi offerti siano stati preparati in modo da girare in un suo compartimento stagno. A meno di azioni particolari dell'utente, quindi, nessun programma dello store potrà accedere a dati riservati o compiere altre operazioni pericolose. Il Click App Store sarà anche in grado di disinstallare a distanza programmi da esso distribuiti, ma poi risultati pericolosi per qualsiasi motivo.

Dal punto di vista degli utenti finali, Click dovrebbe rendere installazione e uso di programmi su un vero sistema Linux mobile molto più simile a quella di Android. Il prezzo da pagare saranno download più lunghi e soprattutto minor libertà, dovendo autolimitarsi a store affidabili e capaci di rimuovere di prepotenza software considerato pericoloso.

In cambio si otterranno programmi che funzionano sicuramente, perché non legati a dipendenze nonché, se provenienti da store affidabili, capaci di segnalare da soli quando hanno bisogno di aggiornamenti. Per gli sviluppatori, usare Click come sistema di packaging e distribuzione dei loro prodotti dovrebbe ridurre fortemente i problemi lamentati da Torvalds, e tanti altri prima di lui, per due motivi. Innanzitutto, una volta configurato l'ambiente di compilazione, la procedura di impacchettamento è più semplice di quella .rpm o .deb. In secondo luogo, creando file eseguibili monolitici, anziché dipendenti da librerie esterne, il programmatore non sarà più "costretto" a rincorrere e studiare tutte le particolarità, aggiornamenti e altre "bizzarrie" del sistema operativo. Certo, se Click diventasse lo

standard per tutte le versioni di Linux, distribuire pacchetti diversi per ogni distribuzione sarebbe ancora necessario, ma richiederebbe meno tempo ed energia della stessa operazione in ambienti .rpm o .deb. I pacchetti Click, infatti, rimangono per loro natura completamente al di fuori delle procedure di installazione e aggiornamento periodico del sistema operativo base, che devono continuare a svolgersi con le procedure tradizionali descritte nell'articolo principale. Lo stesso discorso vale per qualsiasi programma distribuito via repository, anziché dal Click App Store.

FUNZIONERÀ?

Click è sicuramente una delle attività in campo Linux /Open Source da tenere maggiormente d'occhio nel corso del 2015, ma al momento è impossibile fare previsioni sulla sua riuscita. Il programma è ambizioso e l'obiettivo condivisibile, ma dovrà scontrarsi con ostacoli formidabili. Il primo è l'essere agli antipodi o quasi del riuso di librerie e degli altri concetti, ormai dati per scontati su Linux, descritti nell'articolo principale. Progetti come il Desktop Rox o Zero Install sono stati dimenticati da tutti o quasi proprio per questo motivo. Un altro è come far cooperare programmi diversi, magari automaticamente: un dogma nel mondo Linux/Unix, ma molto più difficile da imporre ad applicazioni che girano ognuna sotto una diversa campana di vetro. Anche l'obbligo di dover comunque gestire due flussi di installazione e aggiornamento separati, uno per i programmi Click e l'altro per tutto il resto, potrebbe essere mal digerito da non pochi utenti.

I metadati più essenziali, quelli che hanno davvero iniziato a cambiare la vita degli utenti Linux meno esperti, sono le cosiddette *dipendenze*: dichiarazioni precise di quali versioni di quali *altri pacchetti dello stesso tipo* (non programmi o librerie) occorre aver già installato perché il software nel pacchetto corrente possa funzionare come si deve. Elencare pacchetti anziché software è essenziale perché distribuzioni diverse potrebbero dare nomi leggermente diversi alla stessa libreria. Vale inoltre quanto spiegato a inizio articolo a proposito di ottimizzazione e personalizzazione del software: a parità di nome e versione *del codice sorgente di partenza*, versioni per distribuzioni diverse potrebbero non essere intercambiabili perché installate in cartelle diverse, o compilate con altri parametri.

I GESTORI DI PACCHETTI E REPOSITORIES

Da quanto abbiamo scritto nel paragrafo precedente potrebbe sembrare che l'arrivo di formati come `.deb` o `.rpm`, e dei programmi da riga di comando capaci di gestirli da soli, abbia risolto una volta per tutte la questione dell'installazione software sotto Linux. Un pacchetto ben preparato è installabile con un clic anche se non si

capisce nulla di programmazione e amministrazione di sistema, soprattutto perché può accorgersi da solo se sul computer in cui si trova manca qualcosa e descrivere esattamente di che si tratta.

In realtà, come ben sa come chi abbia provato almeno una volta a svolgere questa attività più di sei o sette anni fa, questa è solo metà del lavoro. Conoscere il nome preciso di tutte le librerie software che mancano per far funzionare un certo programma precompilato può aiutare un utente molto esperto a evitare *tentativi a vuoto*, ma non gli farà certo risparmiare molto tempo. Gli utenti meno esperti si troveranno nella situazione di chi volesse cucinare una torta di cui ha la ricetta, ma senza la minima idea di *dove* comprare alcuni ingredienti o di *chi* potrebbe venderli. Ai file `.rpm` o `.deb` e ai loro installer manca ogni capacità di segnalare all'utente quando è disponibile una nuova versione dello stesso software.

Quest'ultima parte del puzzle viene completata da due pezzi ben distinti, ma fatti l'uno per l'altro: le repositories e i programmi in grado di servirsene.

Le repositories sono archivi online di pacchetti binari, curati dai coordinatori di una certa distribuzione Linux, che ne garantiscono tre caratteristiche fondamentali. La prima è la *specificità*. La repository per Ubuntu 14.04 a 64 bit, tanto per fare un esempio, conterrà *solo* pacchetti `.deb` precompilati per *quella* versione di Ubuntu e per processori a 64 bit, quindi tutti compatibili fra loro e con una installazione base di Ubuntu 14.04.

L'altra garanzia è offerta da una repository è la *completezza*. Un generico pacchetto X contenuto in una certa repository, potrebbe benissimo dipendere da altri venti o trenta pacchetti, contenenti altrettanti programmi, *scritti da altrettanti sviluppatori indipendenti*. Però la repository offrirà quel pacchetto X *solo* se già dispone di quelli di tutti quegli altri programmi. In sostanza, si avrà la garanzia di non avere problemi se ci si limita a installare software da una sola repository (o gruppo di repository, purché compatibili fra loro) specifica della propria distribuzione. L'ultima caratteristica di una repository è il supporto di formati e protocolli standard per dichiarare ai computer che si connettono, quali pacchetti mettono a disposizione e come scaricarli.

Attenti alla distribuzione

Librerie sviluppate per distribuzioni differenti dalla propria potrebbero non essere compatibili

NEWS

COSA TROVEREMO NEL PRIMO UBUNTU DEL 2015

Ubuntu 15.04, la versione della popolare distribuzione Linux che dovrebbe arrivare

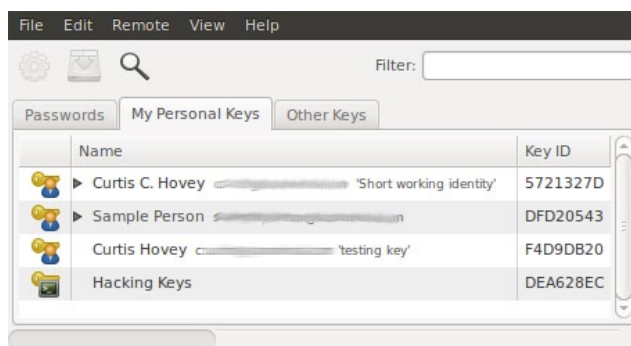
il prossimo aprile, avrà un'interfaccia utente più flessibile. La versione 8 dell'ambiente grafico Unity potrebbe supportare più monitor, finestre di qualsiasi dimensione, rotazione dello schermo e forse un nuovo sistema di notifiche. Dietro le quinte, il sistema di gestione display chiamato Mir dovrebbe poter offrire visualizzazione esterna su schermi Android, ovvero la possibilità di lavorare sul proprio desktop anche da un tablet o smartphone meno potente. Altre novità di tutto rispetto in Mir, se la tabella di marcia verrà rispettata, saranno il supporto per la risoluzione 4K degli schermi televisivi e quello per la composizione di finestre con accelerazione hardware.



In pratica, gli indirizzi e le informazioni sulle repository di base sono già preconfigurate in interfacce grafiche come Apper, Synaptic o Zypper. Per scoprire e installare con pochi clic tutto il software esistente per la propria distribuzione basta imparare a usare uno di questi semplici programmi e la procedura per aggiungervi nuove repository: penseranno loro a scoprire e scaricare da soli tutte le dipendenze necessarie ogni volta che si deciderà di installare qualcosa, nonché a capire quando vanno scaricati aggiornamenti. L'unico vero "lavoro" che l'utente deve ancora fare è scoprire *prima* dell'installazione, cercando su Internet o chiedendo agli altri utenti, quale distribuzione ha le repository più adatte all'uso che si prevede di fare del computer. L'operazione è ovviamente semplificata dai numerosi motori di ricerca in grado di fornire informazioni al riguardo.

L'ALTRA FACCIA DELLA MEDAGLIA: TROPPO LAVORO PER GLI SVILUPPATORI?

Risoluzione automatica delle dipendenze, informazioni dettagliate su ogni pacchetto, compatibilità garantita, massime prestazioni col minimo ingombro, chiusura delle falle di sicurezza aggiornando un solo pacchetto



I Personal Package Archives permettono di distribuire e far installare i pacchetti tramite l'interfaccia standard di Ubuntu a chiunque li prepari e firmi digitalmente secondo le linee guida ufficiali.

anziché N applicazioni... che potrebbe esserci di male in un paradiso del genere? Niente, tranne ciò da cui siamo partiti a inizio articolo, cioè il motivo per cui Linus Torvalds non fornisce pacchetti binari per Linux del *suo* software. Tutti quelli di cui abbiamo parlato sono vantaggi per gli *utenti finali* delle varie distribuzioni. Ma per *dare* a tutti loro software installabile con la stessa facilità di Windows qualcuno dovrebbe preparare e mantenere aggiornato un pacchetto per *ogni* versione di *ogni* distribuzione. Parafrasando Torvalds "non puoi fare pacchetti installabili per Linux. Dovresti farne uno per Fedora 19, uno per Fedora 20... e uno per Debian, se non avesse librerie così vecchie da essere incompatibili con qualsiasi software scritto in questo secolo". Questa dichiarazione, per quanto tecnicamente corretta, tende

un po' al melodrammatico. Tanto per cominciare, nessuno *sviluppatore* è obbligato a fornire personalmente pacchetti dei suoi programmi per tutte le distribuzioni Linux esistenti (diverse decine, anche considerando solo la versione più recente di quelle più attivamente sviluppate). Possono benissimo farlo i *produttori* di quelle distribuzioni, o anche terze parti.

Le distribuzioni più popolari, come Ubuntu o Fedora, ormai hanno comunità talmente grandi e attive da produrre pacchetti praticamente per tutto il software Linux in circolazione. Se invece interessa che i propri programmi siano disponibili per versioni di Linux più specializzate e meno diffuse, come quelle per server o media center, basta creare e mantenere aggiornati pacchetti per queste ultime. •

UN VETERANO DI FEDORA DIVENTA GRANDE: BENVENUTO, NETWORK MANAGER 1.0

Uno dei punti ancora oggi dolenti, per quanto riguarda il supporto e la configurazione dell'hardware sotto Linux, è l'interfaccia per connettersi continuamente a reti diverse, di tipi diversi, da un laptop. Questa situazione potrebbe però migliorare sensibilmente nel 2015, almeno per gli utenti di Fedora e di tutte le altre distribuzioni che gestiscono le connessioni nello stesso modo. La versione 1.0 del Network Manager di Fedora è infatti finalmente arrivata, dopo quasi dieci anni dall'inizio dello sviluppo. Oltre a "innumerevoli", secondo i programmatori, bachi risolti e piccole migliorie, NM 1.0 ha controlli semplificati, supporto per profili multipli ed è molto più integrato con il desktop Gnome dei suoi predecessori.



ENIGMA-DEV, CONCORRENTE LINUX DI GAMEMAKER

Il progetto Enigma-Dev (<http://enigma-dev.org>) produce un ambiente di sviluppo e un motore per giochi Java completamente Open Source. I giochi prodotti possono essere compilati in diverse versioni, per Linux, OS X e Windows, e girare alla massima velocità. Il codice può essere scritto sia in Gml (GameMaker Language, <http://www.yoyogames.com/>), sia in C o C++. L'ambiente Enigma-Dev offre supporto diretto per vari controller e la possibilità di importare giochi in vari formati.