

Debian/GNU z perspektywy administratora (1)

GRZEGORZ JACEK NALEPA

16.3.2000, Kraków, *Revision* : 1.5

Streszczenie

Artykuł jest pierwszym z cyklu opisującego specyfikę zarządzania pakietami w systemie Debian/GNU Linux. Artykuł obejmuje między innymi omówienie struktury dystrybucji Debian/GNU i jej wersji, ogólnej architektury systemu zarządzania pakietami, struktury pakietu DEB, podstaw używania programu `dpkg`. Przedstawione w artykule omówienie ma na celu przybliżenie Debian/GNU administratorom i użytkownikom systemu GNU/Linux.

Spis treści

1	Wstęp	2
2	Struktura dystrybucji i jej wersje	3
2.1	Wersje Debiana i cykl rozwoju dystrybucji	3
2.2	Części dystrybucji	4
2.3	Drzewo plików Debian/GNU	4
3	System zarządzania pakietami	6
4	Struktura pakietu DEB	6
4.1	Nazwy pakietów	6
4.2	Atrybuty pakietów	7
4.3	Zależności	7
4.4	Informacje kontrolne	8
5	Podstawy używania <code>dpkg</code>	8
5.1	Instalowanie pakietów	8
5.2	Odinstalowywanie pakietów	9
5.3	Informacje o pakietach	10
5.4	Inne możliwości	12
6	Podsumowanie	12

¹Tekst ukazał się w: *Magazynie Linux & Unix*, nr 5/2000, wydawanym przez TAO Systems.

²Kontakt z autorem: [mail:gjn@agh.edu.pl](mailto:gjn@agh.edu.pl)

³Tytuł angielski: *Debian/GNU – Administrator's Perspective (1)*

⁴Tekst jest rozpowszechniany na zasadach licencji *GNU Free Documentation License*, której pełny tekst można znaleźć pod adresem: <http://www.gnu.org/copyleft/fdl.html>

1. Wstęp

Debian/GNU Linux jest jedną z najpopularniejszych dystrybucji systemu GNU/Linux. Rozwijany od lat, Debian jest przez wielu uważany za dystrybucję niezwykle stabilną, zaawansowaną technicznie, łatwą w administrowaniu i niezawodną jeżeli chodzi o jej uaktualnianie. Wiele popularnych serwerów linuksowych pracuje na Debianie, przykładem mogą tu być www.linux.com, czy www.linuxnews.pl w Polsce.

Debian/GNU jest jedną z niewielu dystrybucji Linuksa dostępnych na tak wiele platformy, istnieją wersje na procesory Intel x86 (i386), Digital Alpha, Motorola 680x0 (m68k), PowerPC, Sun Sparc i ARM. Dystrybucja składa się z około 4000 tysięcy różnorodnych pakietów oprogramowania, z których wszystkie to oprogramowanie *free software*, oprócz tego dostępny jest obszerny zbiór programów *non-free*.

Warto zauważyć jeszcze jedno. Otóż Debian/GNU jest w tej chwili nie tylko jedną z wielu dystrybucji systemu GNU/Linux. Debian/GNU staje się uniwersalnym systemem operacyjnym, niezależnym od jądra systemu na którym pracuje (na przykład Linuksa). W tej chwili dostępna jest wersja Debiana oparta na jądrze GNU Hurd opracowywanym przez Free Software Foundation, uważanym przez wielu za bardziej zaawansowane technologicznie od Linuksa. Oprócz tego padają propozycje przeniesienia dystrybucji na jądro FreeBSD.

Jest wiele cech odróżniających i wyróżniających Debian/GNU od innych dystrybucji systemu GNU/Linux. Można wspomnieć o stabilności, dużej ilości pakietów oprogramowania, wielu niezwykle użytecznych narzędziach do administrowania, czy chociażby o uniwersalnym systemie menu, sprawiającym, że pod każdym *window managerem* w systemie X Window widoczne są takie same menu, dokładnie odzwierciedlające listę aktualnie zainstalowanych programów. Trzeba również zaznaczyć, że Debian/GNU jako jedyny zawiera wyłącznie oprogramowanie określane jako *free software* (<http://www.debian.org/intro/free>).

To, na które z tych cech kładzie się większy nacisk, zależy od perspektywy z której patrzy się na system. Parząc z perspektywy administratora, cechą wyróżniającą Debian/GNU jest zaawansowany system zarządzania pakietami. Paradoksalnie, to co dla wielu jest największą zaletą, dla innych jest problemem. Debian jako jedyny stworzył prawdziwą alternatywę dla systemu RPM firmy RedHat, używanego w wielu dystrybucjach Linuksa. Jednak ponieważ liczba użytkowników dystrybucji opartych na RPM (takich jak RedHat, Suse, Caldera) jest większa, system RPM jest bardziej popularny i lepiej znany. Dlatego też, często największą przeszkodą dla początkujących administratorów Debiana jest nieznanomość obsługi narzędzi do zarządzania pakietami w tej dystrybucji. Z drugiej strony Ci, którzy je znają, uważają je za jedną z największych zalet dystrybucji.

Trzeba jeszcze przypomnieć o tym, jak bardzo dla administratora ważna jest dokumentacja i wsparcie techniczne. Debian/GNU ma obszerną dokumentację dostępną w dystrybucji i przez strony WWW. Na stronie <http://www.debian.org> użytkownicy mają dostęp do wielu źródeł dokumentacji (<http://www.debian.org/doc>) i wsparcia (<http://www.debian.org/support>), między innymi kilkudziesięciu list dyskusyjnych poświęconych wyłącznie Debianowi i różnym aspektom jego używania i rozwijania (<http://www.debian.org/MailingLists/subscribe>). Polscy administratorzy Debiana, którzy nie znają dobrze angielskiego, znajdują w dystrybucji aktualne polskie tłumaczenia HOWTO z projektu JTZ, system polskich *locale* i inne możliwości lokalizacji.

Jak już wspomniano, z perspektywy administratora, za jeden z najważniejszych i najoryginalniejszych elementów Debian/GNU należy uznać system zarządzania pakietami. Jemu poświęcony jest ten i kolejne artykuły. Omówienie systemu zostało podzielone na następujące części:

- struktura dystrybucji Debian/GNU i jej wersje,

- ogólna architektura systemu zarządzania pakietami,
- struktura pakietu DEB,
- podstawy używania programu `dpkg`,
- zarządzanie pakietami przy pomocy `dselect`,
- konwersje pakietów z pomocą programu `alien`,
- przedstawienie ewolucji systemu pakietów, w tym systemu Apt.

2. Struktura dystrybucji i jej wersje

Przed przystąpieniem do omawiania samego systemu pakietów należy powiedzieć kilka słów na temat struktury dystrybucji Debian/GNU, ponieważ ta struktura jest ściśle związana z systemem pakietów. Ta struktura uwzględnia współistnienie różnych wersji dystrybucji.

2.1. Wersje Debiana i cykl rozwoju dystrybucji

W chwili powstawania tego artykułu, w marcu 2000, wersją *stabilną* Debian/GNU była dystrybucja Debian 2.1 „slink”. Wcześniejsza wersja stabilna, która jest wciąż dostępna to Debian 2.0 „hamm”. Długo oczekiwana nowa wersja – 2.2 „potato” – była wciąż *zamrożona*.¹

W tym miejscu należy powiedzieć o cyklu rozwojowym wersji Debiana. Każda dystrybucja przechodzi przez następujące etapy rozwoju:

1. *unstable* – na początku dystrybucja jest aktywnie rozwijana, w związku z czym jest uważana za *niestabilną*,
2. *frozen* – po kilku miesiącach, kiedy zostaną osiągnięte cele nowej dystrybucji i zintegrowane najważniejsze pakiety oprogramowania, dystrybucja zostaje *zamrożona*. Oznacza to, że nie pojawiają się w niej nowe programy, a wysiłki programistów skupione są na usuwaniu jeszcze obecnych w dystrybucji błędów. Wraz z zamrożeniem dystrybucji, jest tworzona jej kopia, która staje się podstawą nowej dystrybucji niestabilnej.
3. *stable* – dystrybucja zamrożona w której zostaną usunięte wszystkie wykryte błędy, zostaje określona jako *stabilna*. Po pojawieniu się nowej wersji stabilnej zostają tworzone oficjalne obrazy płyt CD-ROM dystrybucji. Wersja stabilna jest aktualizowana, lecz pojawiają się w niej wyłącznie wersje pakietów już w niej istniejących, zawierające łatki dla wykrytych błędów. Każda aktualizacja jest wyraźnie zaznaczana przez numer *release*, pojawiający się po pełnym numerze wersji, na przykład „Debian2.1r5”.

W każdej chwili w drzewie dystrybucji znajduje się przynajmniej wersja stabilna i niestabilna, a w czasie stabilizowania nowej wersji, wszystkie trzy.²

Oprócz określeń oznaczających etap rozwoju każda dystrybucja ma oczywiście własny numer wersji i nazwę, na przykład wersja 2.1 i nazwa „slink”. Ten dualny system nazewnictwa bierze się

¹Wersja Debian/GNU 2.2 ukazała się w sierpniu 2000. W chwili uaktualniania tego artykułu, w kwietniu 2001, wersja stabilna dalej pozostawała 2.2. Kolejna wersja rozwojowa, „woody” zbliżała się do zaplanowanego na lipiec *zamrożenia* i miała być wydana w listopadzie 2001.

²W trakcie prac nad wersją „woody” dodano etap *testing*, będący etapem pośrednim pomiędzy *stable* a *unstable*. W kwietniu 2001 dostępne były następujące wersje dystrybucji: *stable*–„potato”, *testing*–„woody”, *unstable*–„sid”

stać, że Debian/GNU jako jedyny, udostępnia w każdej chwili pełny dostęp do wersji rozwojowej dystrybucji. Użytkownicy i developerzy mogą na bieżąco śledzić rozwój Debiana i w każdym momencie można zainstalować wersję niestabilną, bez czekania na oficjalną wersję stabilną. Ten model rozwoju systemu operacyjnego jest jednym z najlepszych przykładów tego, co za Erykiem S. Raymondem określa się jako „bazaar development”.

2.2. Części dystrybucji

Każda wersja dystrybucji Debian/GNU składa się z czterech podstawowych wersji:

main jest to zasadniczy trzon dystrybucji, zawierający większość pakietów. Wszystkie programy wchodzące w jego skład muszą spełniać *Debian Free Software Guidelines* (DFSG) (http://www.debian.org/social_contract#guidelines), czyli być na licencji GNU GPL, LGPL, lub licencji typu BSD, X Consortium, czy artistic. Tylko część main stanowi „oficjalną dystrybucję Debian/GNU”.

contrib w tej części znajdują się pakiety dodatkowe, które również muszą spełniać DFSG, lecz są przeważnie związane z pakietami non-free, lub pełnią opcjonalne funkcje pomocnicze dla niektórych pakietów z main. Część contrib jest najczęściej dołączana do oficjalnych płyt CD-ROM z oficjalną dystrybucją.

non-free oprogramowanie, które nie spełnia DFSG, jak programy komercyjne (Netscape Navigator, Adobe Acrobat), oprogramowanie chronione patentami (GIF), lub restrykcyjnymi licencjami (Java), znajduje się w części non-free. Ta część jest aktualizowana osobno i nie stanowi integralnej części oficjalnej dystrybucji. Jest dostępna na osobnych płytach CD-ROM i często osobnych serwerach FTP.

non-us w związku z silnymi restrykcjami nakładanymi od lat przez rząd USA na eksport technologii, głównie kryptograficznych, część oprogramowania, które je zawiera, musi znajdować się w osobnej części dystrybucji i na osobnych serwerach i nie jest oficjalnie rozpowszechniane w USA. W tej części można znaleźć chociażby oprogramowanie Ssh. Niewykluczone, że w związku z niedawnym złagodzeniem tych restrykcji, niektóre z programów z non-free znajdują się w innych częściach Debiana.

2.3. Drzewo plików Debian/GNU

Każda z części main, contrib, non-free i non-us zawiera podkatalogi w których są pakiety binarne na różne architektury sprzętowe (*binary-xxx*), pakiety niezależne od architektury (*binary-all*) (zawierające na przykład dokumentację, ikony, programy w językach skryptowych takich jak Perl) i pakiety źródłowe (*source*).

Z kolei wszystkie wymienione powyżej katalogi zawierają pakiety Debiana posortowane na następujące kategorie tematyczne: admin, base, comm, devel, doc, editors, electronics, games, graphics, hamradio, interpreters, libs, mail, math, misc, net, news, oldlibs, otherosfs, science, shells, sound, tex, text, utils, web, x11. Tu warto odnotować, że przy tak dużej ilości pakietów ich sortowanie tematyczne znakomicie ułatwia wyszukiwanie potrzebnych pakietów. W większości innych dystrybucji wszystkie pakiety binarne znajdują się w jednym katalogu (na przykład RPMS).

Po omówieniu podstawowych elementów dystrybucji można zaprezentować w całości drzewo plików Debian/GNU i objaśnić jego elementy na przykładach. Drzewo jest przedstawione na Rysunku 1, który przedstawia stan w marcu 2000. Katalog **stable** jest linkiem do **slink**,

frozen linkiem do **potato**, a **unstable** linkiem do **woody**. Jak już wspomniano, wraz z pojawieniem się nowej wersji stabilnej, katalog (link) **stable** będzie wskazywał na katalog **potato**, katalog **frozen** zniknie aż do momentu stabilizowania wersji „woody”, a dystrybucja „slink” będzie dostępna wyłącznie w katalogu **slink**.

```
debian
  doc
  project
  tools
  dists
    stable
    frozen
    unstable
    woody
    potato
    slink
  contrib
  non-free
  non-us
  main
    binary-all
    binary-i386
    binary-m68k
    binary-alpha
  admin
  base
  ...
  mail
  math
  ...
  web
  x11
  disks-i386
  disks-m68k
  disks-alpha
  source
```

Rysunek 1: Drzewo plików Debian/GNU

Katalogi `/debian/dists/xxxx/main`, gdzie `xxxx` oznacza nazwę dystrybucji, na przykład **slink**, zawierają pliki **Contents-yyyy**, gdzie `yyyy` oznacza architekturę, na przykład `-i386`. Te pliki zawierają spis wszystkich plików, które są w pakietach DEB w tej wersji dystrybucji na danej platformie sprzętowej. Przykład: `/debian/dists/stable/main/Contents-i386`.

Podkatalogi **binary-xxx** zawierają pliki **Packages**, będące spisem wszystkich pakietów DEB znajdujących się w danej wersji (na przykład „slink”) i części (na przykład **main**) dystrybucji na danej platformie (na przykład **i386**). Przykład:

`/debian/dists/stable/main/binary-i386/Packages`.

Korzystając z Rysunku 1, można na przykład zauważyć, że pliki binarne wersji stabilnej na platformę Intel x86 można znaleźć w katalogach: `/debian/dists/stable/main/binary-i386` i `/debian/dists/stable/main/binary-all`. Obrazy dyskielek startowych, konieczne do instalacji, są w katalogu `/debian/dists/stable/main/disks-i386`. Pakiety źródłowe są w katalogu `/debian/dists/stable/main/source`. Ponieważ w chwili pisania artykułu, wersją stabilną był „slink”, te same katalogi były dostępne, jeżeli część **stable** w podanych ścieżkach dostępu zastąpiło się przez **slink**.

Wersje na inne platformy sprzętowe znajdują się w katalogach, w których końcówkę `-i386`, zastąpi się przez inną, na przykład `-m68k` dla platformy Motorola 680x0, lub `-sparc` dla platformy Sun Sparc.

Na uwagę zasługuje katalog *binary-hurd-i386*, występujący w dystrybucji Woody, zawiera-

jący wersję Debian/GNU na platformę Intel x86 opartą nie o jądro Linux, lecz o jądro systemu GNU Hurd (<http://www.gnu.org/software/hurd/hurd.html>), pisanego przez Free Software Foundation.

Znając ogólną strukturę dystrybucji można przejść do omawiania systemu zarządzania pakietami i struktury pakietów DEB.

3. System zarządzania pakietami

System zarządzania pakietami w Debian/GNU jest złożony i składa się z kilku narzędzi. Wszystkie z tych narzędzi zapewniają interfejsy do pewnych części bazy danych systemu pakietów. Przed omówieniem tych narzędzi trzeba zarysować ogólną architekturę tej bazy danych.

Baza danych systemu pakietów przechowuje między innymi następujące informacje:

- spis zainstalowanych pakietów,
- spis dostępnych pakietów,
- dane kontrolne pakietów.

Na dokładne omówienie tych elementów będzie miejsce dalej, przy okazji omawiania struktury samego pakietu DEB, oraz narzędzia `dselect`.

Istnieje szereg narzędzi zarządzających pakietami, są to:

dpkg najważniejszy program, przy pomocy niego można przede wszystkim instalować i odinstalowywać pakiety. Umożliwia również uzyskiwanie informacji o zainstalowanych pakietach, oraz aktualizowanie bazy danych o pakietach.

dselect jest graficznym interfejsem pracującym na konsoli, umożliwiającym wizualizację informacji o pakietach. Pozwala na przeszukiwanie i sortowanie bazy danych, wybieranie grup pakietów do instalacji i automatyczne sprawdzanie zależności.

apt jest nowym interfejsem do systemu pakietów, który ma być w pełni dostępny w dystrybucji Potato. Ma bardzo wiele zaawansowanym funkcji.

dpkg-dev **debmake** te i kilka innych narzędzi służą do operowania na samych plikach pakietów, w tym do budowania pakietów źródłowych.

Najważniejszym z tych narzędzi jest **dpkg**. Dla użytkowników dystrybucji opartych o RPM, można go porównać do samego programu **rpm**, choć jak będzie opisane dalej, funkcjonalność tych dwóch narzędzi jest nieco inna. W przyszłości, już w dystrybucji Potato, coraz bardziej będą się liczyły narzędzia z rodziny Apt.

Aby zrozumieć sposób działania **dpkg** i proces instalowania pakietów, koniecznym wydaje się przedstawienie struktury pakietu DEB.

4. Struktura pakietu DEB

4.1. Nazwy pakietów

Pakiet DEB jest archiwum typu Ar (ar(1)). Pełna nazwa pliku pakietu ma postać:

`nazwapakietu_wersjaprogramu-wersjapakietu.deb`

Na przykład:

`bash_2.01.1-4.1.deb`

oznacza pakiet `bash`, zawierający program `bash`, w wersji 2.01. Liczba 4.1 oznacza wersję samego pakietu `bash`. Wersje pakietów oznaczają zmiany w samym pakiecie DEB wprowadzane przez członka projektu Debian, który zajmuje się pakietem.

Nazwa pakietu czasami zawiera przyrostki `-dev` czy `-doc`. Niektóre programy, dotyczy to na przykład pewnych bibliotek, są dzielone na kilka pakietów. Na przykład biblioteka Gtk+ jest dzielona na trzy pakiety: `libgtk1.2`, `libgtk1.2-dev` i `libgtk1.2-doc`. Pierwszy z nich zawiera binaria biblioteki, potrzebne do pracy skompilowanych z nią programów (ang. *runtime*), w drugim są pliki potrzebne do kompilowania programów z tą biblioteką, ostatni zawiera tylko dokumentację do biblioteki. Dzięki temu, przeciętnemu użytkownikowi wystarczy tylko pierwszy pakiet, a programista może wykorzystać dwa pozostałe. Pozwala to na zaoszczędzenie miejsca w systemie plików.

4.2. Atrybuty pakietów

W systemie Debian/GNU pakiety są pogrupowane w drzewie plików na części omówione już powyżej, to znaczy *części dystrybucji* (na przykład `main`) i *kategorie tematyczne* (na przykład `admin`, `net`).

Oprócz tego pakiety mają *priorytety*. Pakiet może mieć jeden z następujących priorytetów:

required (wymagany) oznacza, że pakiet jest niezbędny do funkcjonowania systemu, musi być zainstalowany i nie można go usuwać.

important (ważny) wskazuje, że pakiet powinien być zawsze zainstalowany by zapewniać podstawową funkcjonalność systemu typu Un*x (nie obejmuje to bynajmniej dużych pakietów, typu Emacs, X11).

standard (standardowy) zainstalowanie pakietów o tym priorytecie pozwala na stworzenie podstawowego uniwersalnego systemu pracującego w trybie znakowym. Przeważnie większość pakietów o tym priorytecie jest zainstalowana.

optional (opcjonalny) w ten sposób oznaczone są pakiety, które wielu użytkowników może chcieć zainstalować w systemie, w tej kategorii mieści się chociażby system X Window. Wybór pakietów o tym priorytecie jest arbitralny i zależy od ogólnych zastosowań systemu.

extra (dodatkowe) specjalizowane pakiety do konkretnych zastosowań, ich wybór jest ściśle podyktowany specjalnymi funkcjami jakie ma pełnić system.

Wszystkie informacje dotyczące kategorii pakietów, ich priorytetów i ich podziału na grupy można znaleźć w dokumencie Debian-Policy (<http://www.debian.org/doc/debian-policy>).

4.3. Zależności

Zachowanie spójności dystrybucji nie byłoby możliwe bez określenia zależności pomiędzy pakietami (ang. *dependency*). Zależności mogą być wielu rodzajów, wszystkie będą omówione przy okazji `dselect`. W tej chwili warto wspomnieć przynajmniej o poniższych.

Pakiet X może wymagać pakietu Y (ang. **depends**), to znaczy, że do działania X, konieczne jest zainstalowanie Y. Z drugiej strony pakiet X może być w konflikcie z pakietem Y (ang. **conflicts**), to znaczy, że X nie może być równocześnie zainstalowany z Y i tylko jeden z nich może być zainstalowany.

4.4. Informacje kontrolne

Pakiet DEB oprócz oprogramowania, którego wersję instalacyjną zawiera, jest również wyposażony w informacje kontrolne. Są to na przykład informacje o pakiecie, takie jak jego atrybuty, opis, czy autor. Oprócz tego znajdują się w nim również skrypty konfiguracyjne, które umożliwiają prawidłowe skonfigurowanie pakietu w systemie. Mogą one być uruchamiane przed i po instalacji czy usunięciu pakietu. Dzięki temu mechanizmowi, w systemie Debian/GNU możliwe jest uaktualnianie pakietów w trakcie działania poprzednich wersji programów, ponieważ skrypty instalacyjne automatycznie zatrzymają i uruchomią odpowiednie programy.

Dokładniejsze omówienie informacji kontrolnych pakietu DEB znajduje się w części opisującej pakiety źródłowe DEB.

Po przedstawieniu architektury systemu pakietów można przejść do omawiania najważniejszych narzędzi, dających do niego dostęp.

5. Podstawy używania dpkg

Program **dpkg** jest od dawna podstawowym narzędziem do zarządzania pakietami w systemie Debian/GNU. Jest obsługiwany z linii poleceń.

dpkg ma następujące możliwości:

- instalowanie (uaktualnianie) i odinstalowywanie pakietów,
- uzyskiwanie informacji o zainstalowanych pakietach,
- aktualizację bazy danych o pakietach, oraz inne operacje na samej bazie,
- operacje na plikach DEB (przy pomocy **depkg-deb**).

Program **dpkg** nie potrzebuje osobnej opcji do dokonywania aktualizacji pakietu. Procedura instalowania jest uniwersalna i wykrywając poprzednią wersję pakietu, umożliwia jego automatyczne uaktualnienie.

5.1. Instalowanie pakietów

Instalację pakietu DEB przy pomocy **dpkg** przeprowadza się następująco:

```
dpkg -i nazwa_pliku_pakietu.deb
```

Należy podać nazwę pliku, wraz z pełną ścieżką dostępu.

Instalacja (uaktualnienie) pakietu przebiega następująco:

1. z pliku DEB są wypakowywane informacje kontrolne o pakiecie,
2. sprawdzane są zależności,
3. jeżeli jest zainstalowana poprzednia wersja pakietu o tej nazwie, uruchamiany jest skrypt **prerm** poprzedniego pakietu,
4. uruchamiany jest skrypt **preinst** instalowanego pakietu,
5. pliki nowego pakietu są rozpakowywane do systemu plików, równocześnie wykonywane są kopie zapasowe plików poprzedniej wersji (o ile taka jest),

6. jeżeli była zainstalowana poprzednia wersja pakietu o tej nazwie, uruchamiany jest skrypt `postrm` poprzedniego pakietu,
7. rozpakowywane są pliki konfiguracyjne nowego pakietu; jeżeli jest poprzednia wersja pakietu, najpierw wykonywane są kopie zapasowe starych plików,
8. uruchamiany jest skrypt `postinst` nowo zainstalowanego pakietu,
9. jeżeli instalacja (uaktualnienie) przebiegła pomyślnie, to w razie potrzeby są usuwane robocze kopie zapasowe plików poprzedniej wersji pakietu, ale bez usuwania kopii plików konfiguracyjnych.

Wspomniane skrypty: `preinst`, `postinst`, `prerm` i `postrm` są uruchamiane odpowiednio: przed instalacją (ang. *pre installation*), po instalacji (ang. *post installation*), przed usunięciem (ang. *pre removal*) i po usunięciu (ang. *post removal*) pakietu. Dokonują one automatycznej konfiguracji lub dekonfiguracji pakietu. W przypadku mniej złożonych pakietów, nie wszystkie z tych skryptów muszą występować.

W praktyce, administrator może nie wiedzieć o tych wszystkich etapach, gdyż `dpkg` informuje tylko o ogólnym przebiegu instalacji, to znaczy o rozpakowywaniu plików, konfigurowaniu pakietu, lub uaktualnianiu jego starej wersji. Tym niemniej, poznanie całości procesu instalacji ułatwia jego zrozumienie.

Jak widać, procedura instalacji jest złożona, lecz dzięki temu umożliwia bezproblemowe automatyczne uaktualnienie pakietu, a w razie problemów przywrócenie jego starej wersji. Dzięki tej procedurze instalacji i uaktualniania Debian/GNU jest uważany za bardzo niezawodny, jeżeli chodzi o zarządzanie pakietami.

Jest kilka przydatnych opcji, które można podać `dpkg`, obok opcji `-i`, przed instalacją (i ewentualnie deinstalacją):

- `E` – zapobiega instalacji pakietów o wersji takiej samej, jak pakiet zainstalowany,
- `G` – zapobiega instalacji pakietów o wersji niższej, niż pakiet zainstalowany,
- `R` – rekursywnie instaluje pakiety z podanego katalogu.

Może się zdarzyć, że w trakcie instalacji pakietu trzeba doinstalować inne pakiety, których wymaga (ang. *depends*) instalowany pakiet (na przykład biblioteki). Robi się to uruchamiając `dpkg -i` i podając nazwy plików wymaganych pakietów. W takim przypadku, pakiet instalowany jako pierwszy nie został w pełni zainstalowany, a konkretnie skonfigurowany (dlatego, że nie były zainstalowane pakiety, których wymagał). Konieczna jest w takim przypadku ręczna konfiguracja pakietu.

```
dpkg --configure nazwa_pakietu
```

Aby skonfigurować wszystkie nieskonfigurowane pakiety można napisać:

```
dpkg --configure -a
```

5.2. Odinstalowywanie pakietów

Pakiety odinstalowuje się przy pomocy opcji `-r` (`--remove`).

```
dpkg -r nazwa_pakietu
```

Podaje się przy tym nazwę pakietu (bez numerów wersji), a nie nazwę pliku pakietu.

Ten sposób instalacji nie usuwa plików konfiguracyjnych pakietu, gdyż mogły one być modyfikowane przez administratora. Jeżeli jednak chce się usunąć również te pliki w trakcie odinstalowywania, to można użyć opcji **--purge** zamiast opcji **-r**.

Odinstalowywanie pakietu składa się z poniższych etapów:

1. Sprawdzenie zależności pakietu. Jeżeli odinstalowanie pakietu mogłoby naruszyć działanie innych pakietów, które od niego zależą, pakiet nie zostanie odinstalowany. Można rzecz jasna wymusić odinstalowanie pakietu, korzystając z opcji **-B**, lub opcji **--force-depends** (ewentualnie **--ignore-depends**).
2. Uruchomienie skryptu **prerm** (*preremoval*).
3. Usunięcie plików pakietu, jeżeli użyto **--purge**, to również plików konfiguracyjnych.
4. Uruchomienie skryptu **postrm** (*postremoval*).

W trakcie instalacji, zaawansowany użytkownik może wymusić pewne działania, przed którymi ostrzega **dpkg** (na przykład weryfikowanie zależności). Służą do tego opcje z grupy **--force**. Umożliwiają one między innymi ignorowanie zależności pakietów (**--force-depends**), czy ignorowanie platformy na którą jest przeznaczony pakiet (**--force-architecture**). Pełny opis tych i innych opcji **dpkg** można znaleźć w **dpkg(8)**.

5.3. Informacje o pakietach

Jedną z ważnych funkcji **dpkg** jest pokazywanie informacji o zainstalowanych pakietach.

Stan pakietu, wraz ze zwięzłym opisem i numerem wersji można uzyskać przy pomocy polecenia:

```
dpkg -l nazwa_pakietu
```

Użycie powyższego polecenia bez podanej nazwy pakietu spowoduje wyświetlenie informacji o wszystkich zainstalowanych pakietach. Na przykład:

```
$ dpkg -l bash
Desired=Unknown/Install/Remove/Purge
| Status=Not/Installed/Config-files/Unpacked/Failed-config/Half-installed
|/ Err?=(none)/Hold/Reinst-required/X=both-problems (Status,Err: uppercase=bad)
||/ Name          Version          Description
++-=====
ii  bash           2.01.1-4.1       The GNU Bourne Again SHell
```

Jeżeli chce się uzyskać szczegółowe informacje o zainstalowanym pakiecie należy użyć opcji **-s** (**status**):

```
dpkg -s nazwa_pakietu
```

Na przykład:

```
$ dpkg -s bash
Package: bash
Essential: yes
Status: install ok installed
Priority: required
```

```
Section: base
Installed-Size: 783
Maintainer: Guy Maor <maor@debian.org>
Version: 2.01.1-4.1
Pre-Depends: libc6 (>= 2.0.7u), libncurses4, libreadline2 (>= 2.1-8)
Conffiles:
  /etc/profile db431f071e29c47666be4117325e641d
  /etc/skel/.bash_profile 7fb37112e03e428ceb7755b36bd08a42
  /etc/skel/.bashrc 118f882acfaf472e6a352b1baaf5db87
Description: The GNU Bourne Again SHell
  Bash is an sh-compatible command language interpreter that executes
  commands read from the standard input or from a file. Bash also
  incorporates useful features from the Korn and C shells (ksh and csh).
  .
  Bash is ultimately intended to be a conformant implementation of the
  IEEE Posix Shell and Tools specification (IEEE Working Group 1003.2).
```

Na powyższym przykładzie widać dodatkowo, że pakiet `bash` należy do grupy pakietów niezbędnych do działania systemu (ang. *essential*).

W sytuacji, gdy chce się sprawdzić do jakiego pakietu należy jakiś plik, można użyć polecenia:

```
dpkg -S nazwa_pliku
```

Na przykład:

```
$ dpkg -S /etc/profile
bash: /etc/profile
```

Często, administratora interesuje zawartość pakietu, to znaczy lista plików, które wchodzi w jego skład. Wyświetla się ją przy pomocy:

```
dpkg -L nazwa_pakietu
```

Na przykład:

```
$ dpkg -L hdparm
/.
/usr
/usr/sbin
/usr/sbin/hdparm
/usr/man
/usr/man/man8
/usr/man/man8/hdparm.8.gz
/usr/doc
/usr/doc/hdparm
/usr/doc/hdparm/README.Debian
/usr/doc/hdparm/copyright
/usr/doc/hdparm/changelog.Debian.gz
```

5.4. Inne możliwości

System pakietów ma bazę danych na temat pakietów, które są dostępne, to znaczy znajdują się w zdefiniowanym miejscu, na przykład na instalacyjnej płycie CD-ROM, lub serwerze NFS, FTP.

Polecenie `dpkg` ma szereg opcji dotyczących aktualizacji bazy danych o dostępnych pakietach:

- `--update-avail Packages` – buduje nową bazę na podstawie podanego pliku `Packages` (taki plik znajduje się standardowo w każdym katalogu typu `binary` – w drzewie plików Debiana),
- `--merge-avail Packages` – dopisuje do bazy pakiety z pliku `Packages`,
- `--clear-avail` – czyści zawartość bazy.

Te opcje łatwiej jest zrozumieć podczas używania `dselect`, co będzie omówione dalej.

Oprócz omówionych powyżej opcji, `dpkg` zapewnia interfejs do polecenia `dpkg-deb` operującego na samych plikach pakietów. Nie są to opcje często używane przez przeciętnego administratora, zainteresowani mogą znaleźć ich opis w `dpkg-deb(1)`.

Spis wszystkich opcji `dpkg` można w każdej chwili oglądać przy pomocy polecenia `dpkg -h`, a ich szczegółowy opis znajduje się rzecz jasna w podręczniku `dpkg(8)`. Podsumowanie najważniejszych z omówionych opcji można znaleźć w Tabeli 1.

Opcja	Argument	Opis
<code>-i</code>	<code>nazwa_pliku_pakietu.deb</code>	Instalacja pakietu
<code>--configure</code>	<code>nazwa_pakietu</code>	Konfiguracja pakietu
<code>-r</code>	<code>nazwa_pakietu</code>	Usunięcie pakietu
<code>--purge</code>	<code>nazwa_pakietu</code>	Usunięcie pakietu i jego plików konfiguracyjnych
<code>-l</code>	<code>[nazwa_pakietu]</code>	Stan wszystkich zainstalowanych pakietów (lub podanego)
<code>-L</code>	<code>nazwa_pakietu</code>	Spis plików w zainstalowanym pakiecie
<code>-s</code>	<code>nazwa_pakietu</code>	Dokładny opis zainstalowanego pakietu
<code>-S</code>	<code>nazwa_pliku</code>	Odnalezienie pakietu do którego należy podany plik

Tablica 1: Opcje `dpkg`

6. Podsumowanie

System zarządzania pakietami w dystrybucji Debian/GNU jest złożony, lecz niezwykle uniwersalny i wysoce niezawodny, co zauważa i docenia wielu administratorów.

Niejednokrotnie spotykane opinie, jakoby system pakietów Debian/GNU miał być trudniejszy w obsłudze niż popularny RPM, wydają się nieuprawnione. Opinie te należy raczej złożyć na karb nieco mniejszej popularności dystrybucji Debian/GNU.

Przedstawione w artykule omówienie ma na celu przybliżenie Debian/GNU administratorom i użytkownikom systemu GNU/Linux. Omówienie nie byłoby pełne bez omówienia narzędzi `dselect`, `alien` i `apt`. Programy te zostaną przedstawione w kolejnych artykułach.