

# Debian/GNU z perspektywy administratora (3)

GRZEGORZ JACEK NALEPA

16.5.2000, Kraków, *Revision* : 1.6

---

## Streszczenie

Artykuł jest trzecim i ostatnim z cyklu opisującego specyfikę zarządzania pakietami w systemie Debian/GNU Linux. W artykule jest przedstawione narzędzie Alien pozwalające na instalowanie pakietów innych dystrybucji oraz system APT będący następną generacją narzędzi do zarządzania pakietami w systemie operacyjnym Debian/GNU.

---

## Spis treści

<b>1</b>	<b>Nowy system pakietów</b>	<b>2</b>
<b>2</b>	<b>System APT</b>	<b>2</b>
2.1	Co nowego w Potato . . . . .	3
2.2	Rodzina narzędzi APT . . . . .	3
2.3	apt-get . . . . .	4
2.3.1	Konfigurowanie . . . . .	4
2.3.2	Wywoływanie . . . . .	4
2.4	Rekompilacja pakietów źródłowych . . . . .	6
2.5	APT i Dselect . . . . .	6
2.6	Konfigurowanie APT . . . . .	7
2.7	Dokumentacja APT . . . . .	7
<b>3</b>	<b>Alien</b>	<b>7</b>
3.1	Funkcje i sposób użycia . . . . .	8
<b>4</b>	<b>Architektura systemu pakietów</b>	<b>9</b>
<b>5</b>	<b>Podsumowanie</b>	<b>10</b>

---

<sup>1</sup>Tekst ukazał się w: *Magazynie Linux & Unix*, nr 7/2000, wydawanym przez TAO Systems.

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

<sup>3</sup>Tytuł angielski: *Debian/GNU – Administrator's Perspective (3)*

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

Poprzednie artykuły omawiały strukturę systemu pakietów w Debian/GNU oraz podstawowe narzędzia będące jego częścią. Ostatnia część opisuje system APT, będący kolejnym etapem rozwoju Debiana. W następnych wersjach dystrybucji Debian, APT będzie odgrywał coraz większą rolę. Najnowsza dystrybucja,<sup>1</sup> Debian/GNU 2.2 „Potato”, jest już w pełni zintegrowana z APT, tak więc znajomość jego działania jest niezbędna w administrowaniu. Na zakończenie zostanie również przedstawiony program *alien*, będący użytecznym rozszerzeniem podstawowych narzędzi.

## 1. Nowy system pakietów

Na podstawie zawartego w poprzednich częściach omówienia systemu pakietów Debian/GNU można powiedzieć, że jest to system złożony, lecz uniwersalny. Tym niemniej, wraz z upływem czasu i rozwojem systemu Debian/GNU, a także gwałtownym rozwojem technologii pojawiła się potrzeba rozbudowy systemu pakietów.

System oparty na programach *dpkg* i *dselect* dobrze spełnia swoje zadania w sytuacjach, gdy instalowane pakiety pochodzą z jednego źródła, jakim jest na przykład płyta CD-ROM, czy pojedynczy katalog na serwerze FTP. Tymczasem współcześnie coraz częściej mamy do czynienia z sytuacjami, w których oprogramowanie jest aktualizowane z kilku źródeł, takich jak serwery FTP, czy HTTP. Te źródła uzupełniają wersję instalacyjną systemu o nowsze wersje pakietów, lub nowe pakiety.

Wykorzystywanie wielu źródeł pakietów równocześnie może w praktyce powodować różne problemy. Przykładowo, administrator może wykorzystywać dwa różne serwery FTP zawierające pakiety Debiana, z których jeden zawiera kompletną kopię całej dystrybucji, a drugi jej fragment. Pomimo, że pierwszy serwer jest bardziej kompletny, to jest rzadziej uaktualniany. Drugi z kolei zawiera jedynie część dystrybucji, lecz bardzo często uaktualnianą. Powiedzmy, że dodatkowo na maszynie są używane środowiska GNOME czy KDE, których najnowsze wersje są w pakietach na oddzielnych serwerach. W przedstawionym przykładzie pojawia się kilka problemów, sprowadzających się w praktyce do zachowania *spójności* bazy danych o dostępnych pakietach. System pakietów powinien być w stanie połączyć informacje o pakietach dostępnych na czterech różnych serwerach i dołączyć je do bazy pakietów dostępnych na dysku lokalnym, czy płycie CD-ROM. Następnie powinien być w stanie odnaleźć żądany pakiet w jednym z powyższych miejsc i po ustaleniu jego zależności odnaleźć, o ile to możliwe pakiety, konieczne do spełnienia tych zależności.

Przedstawione powyżej zadanie nie było w praktyce realizowane przez żaden system pakietów w dystrybucjach systemu GNU/Linux. Najczęściej zadanie odnalezienia odpowiedniej wersji właściwego pakietu spadało na administratora. Rzecz jasna powstało szereg narzędzi wspomagających to wyszukiwanie, najczęściej serwisów na stronach WWW. Niestety żadne z tych rozwiązań nie potrafiło zachowywać automatycznie pełnej spójności informacji o pakietach zainstalowanych i dostępnych.

## 2. System APT

Na początku roku 1998 w ramach projektu Debian rozpoczęto prace nad systemem zarządzania pakietami rozwiązującym opisane powyżej problemy. System APT (ang. *Advanced Package Tool*), o którym jest mowa, jest rozwiązaniem ewolucyjnym. Nie zastępuje dotychczasowych narzędzi takich jak *dpkg*, ale rozszerza ich możliwości.

---

<sup>1</sup>Najnowsza, w chwili pisania artykułu. Pojawienie się kolejnej wersji, „Woody”, jest planowane na listopad 2001.

Podstawowym zadaniem systemu APT jest automatyzacja procesu wyszukiwania, aktualizacji, instalowania i odinstalowywania pakietów DEB, przy równoczesnym zapewnieniu spójności informacji o pakietach dostępnych i zainstalowanych. System kontroluje na bieżąco zależności wszystkich zainstalowanych pakietów. Umożliwia również automatyczną rekompilację pakietów źródłowych.

Na najniższym poziomie, to znaczy na etapie samego instalowania pakietów, po ich uprzednim odnalezieniu i jeżeli jest taka potrzeba skopiowaniu do lokalnego systemu plików, APT wykorzystuje `dpkg`. APT nie ma zastępować `dpkg`, ma za to zastąpić `dselect`. W przeciwieństwie do `dselect`, APT ma zapewniać szereg różnorodnych interfejsów użytkownika umożliwiających zarządzanie pakietami. Rozwijanych jest kilka tego typu interfejsów, takich jak pracujące w trybie tekstowym `apt-get`, `console-apt` i `aptitude`, oraz pracujące w trybie graficznym `dietty` i `gnome-apt`. Mają one spełniać różne funkcje i nie jest w tej chwili pewne, które z nich zostaną ostatecznie ukończone, a które zostaną połączone z innymi. Dlatego też warto się skupić na aktualnym stanie systemu APT.

## 2.1. Co nowego w Potato

Z punktu widzenia obecnych i przyszłych administratorów systemów Debian/GNU istotne jest, jakie narzędzia mają do dyspozycji. APT, a właściwie jego skromna część `apt-get`, pojawił się po raz pierwszy w Debian/GNU 2.1 „Slink”. Umożliwiał on przede wszystkim bezproblemowe uaktualnienie dystrybucji oraz znajdowanie w sieci odpowiednich wersji pakietów. Jednak dopiero wersja Debian/GNU 2.2 „Potato” zawiera większą część APT. Dalej zostaną omówione te narzędzia, które są dostępne w „Potato”.

## 2.2. Rodzina narzędzi APT

W „Potato” jest dostępny APT w wersji 0.3.18. Nie zawiera on jeszcze wspomnianych zaawansowanych interfejsów użytkownika. Realizuje natomiast przedstawione w punkcie 1 założenia dotyczące zarządzania pakietami. Składa się on z następujących programów:

- `apt-get` – jest to podstawowe narzędzie służące do pobierania pakietów z dowolnych podanych źródeł.
- `apt-config`, `apt-setup` – są to narzędzia służące do konfiguracji systemu APT i programu `apt-get`.
- `apt-cache` – umożliwia bezpośredni dostęp do informacji o wszystkich dostępnych i zainstalowanych pakietach (ang. *package-cache*). Może być wykorzystywany między innymi do wyszukiwania pakietów w bazie APT.
- `apt-cdrom` – służy do dodawania do bazy dostępnych pakietów informacji o pakietach na płytach CD-ROM.
- `apt-move` – pozwala na synchronizację baz danych o pakietach pomiędzy różnymi maszynami.
- `apt-zip` – ułatwia pracę APT z wymiennymi nośnikami takimi jak dyskietki, napędy ZIP i inne.

Wymienione narzędzia wykorzystują bibliotekę `libapt-pkg`, zawierającą najważniejsze funkcje tych programów. Najważniejszą funkcję spełnia `apt-get` i jemu warto poświęcić najwięcej uwagi.

## 2.3. apt-get

Narzędzie **apt-get** zapewnia prosty interfejs do systemu APT umożliwiający przede wszystkim instalowanie pakietów. Program nie operuje na plikach DEB, lecz na samych pakietach. W związku z tym, **apt-get** nie instaluje plików DEB znajdujących się w systemie plików, lecz pakiety, znajdujące się w jednym z podanych źródeł. Dostępne źródła pakietów są podawane podczas konfiguracji.

### 2.3.1. Konfigurowanie

Konfiguracja **apt-get** znajduje się w pliku `/etc/apt/sources.list`. Plik `sources.list`(5) składa się z linii, z których każda ma postać:

typ URI argumenty

Parametr **typ** może obecnie przyjmować jedną wartość `-deb` i oznacza typowe drzewo dystrybucji Debiana, mające strukturę typu: `dists/dystrybucja/część_dystrybucji`.

URI (ang. *Universal Resource Identifier*) jest nadzbiorem popularnych URL, w związku z czym ma do nich podobny format. Przykłady URI:

```
http://www.debian.org/archive
ftp://debian.mps.krakow.pl/mirror/debian
cdrom:Debian 2.2/debian
file:/var/debian
```

Dodatkowe argumenty zawierają informacje o wykorzystywanych częściach dystrybucji.

W związku z tym przykład kompletnej, poprawnej linii w pliku `sources.list` może być następujący:

```
deb ftp://debian.mps.krakow.pl/mirror/debian potato main contrib
```

Oznacza ona, że **apt-get** będzie pobierał pakiety z maszyny `debian.mps.krakow.pl`, przez protokół FTP z katalogu `/mirror/debian`. Zostaną pobrane informacje o pakietach z dystrybucji `potato`, z części `main` i `contrib`.

Konfigurację wygodnie jest przeprowadzić przy pomocy interfejsu **apt-setup**. Umożliwia on szybkie i półautomatyczne wygenerowanie pliku `sources.list`. Wystarczy odpowiedzieć na pytania dotyczące lokalizacji w sieci i części dystrybucji jakie będą używane. Program sam zaproponuje serwery FTP dostępne w danym kraju, na przykład po podaniu Polski można przykładowo wybrać serwer `debian.mps.krakow.pl`, lub `ftp.icm.edu.pl`.

### 2.3.2. Wywoływanie

Po skonfigurowaniu można wywoływać **apt-get** w sposób następujący:

```
apt-get [opcje] [polecenie] [pakiet]
```

Najważniejsze polecenia to:

- **update** – powoduje uaktualnienie informacji o pakietach dostępnych w źródłach wymienionych w pliku `sources.list`.
- **upgrade** – dokonuje automatycznego uaktualnienia całego systemu. Program porównuje wersje zainstalowanych pakietów z wersjami dostępnymi i w razie potrzeby instaluje nowsze wersje, sprawdzając wszystkie zależności. To polecenie może służyć do systematycznego, zautomatyzowanego uaktualniania systemu.

- **install** – wraz z podanymi nazwami pakietów powoduje odnalezienie ich najnowszej wersji, oraz jej zainstalowanie.
- **dist-upgrade** – dokonuje uaktualnienia pomiędzy dystrybucjami Debian/GNU pozwalając przykładowo na automatyczne uaktualnienie dystrybucji 2.1 do 2.2.
- **remove** – usuwa podany pakiet z systemu.
- **check** – sprawdza stan systemu pakietów wykrywając i raportując ewentualne niespójności i inne błędy.
- **clean** – usuwa pliki pakietów DEB, które były skopiowane z jednego z podanych źródeł przed zainstalowaniem.
- **source** – pozwala na uzyskanie pakietu źródłowego DEB, jego automatyczną rekompilację i zainstalowanie.

Poniżej pokazany jest przykład instalowania pakietu przy pomocy **apt-get**:

```
# apt-get install task-polish
Reading Package Lists... Done
Building Dependency Tree... Done
The following extra packages will be installed:
  doc-linux-pl fonty ipolish konwert konwert-filters manpages-pl wpolish
The following NEW packages will be installed:
  doc-linux-pl fonty ipolish konwert konwert-filters manpages-pl task-polish
0 packages upgraded, 7 newly installed, 0 to remove and 0 not upgraded.
Need to get 5492kB of archives. After unpacking 15.0MB will be used.

Do you want to continue? [Y/n]

Get:1 ftp://debian.mps.krakow.pl potato/main doc-linux-pl...
...
Selecting previously deselected package...
Unpacking... (from ...) ...
Setting up...
```

Jak widać, przed przystąpieniem do instalowania jest budowana baza o wszystkich dostępnych pakietach, oraz drzewo zależności pakietów. Następnie są sprawdzane zależności. W tym przypadku instalowany jest pakiet **task-polish** zawierający oprogramowanie, które w dużym stopniu umożliwi spolszczenie Debiana. Ten pakiet wymaga szeregu innych, takich jak czcionki, konwertery, czy polski słownik. Po ustaleniu zależności i listy instalowanych pakietów jest wyświetlane podsumowanie. Zawiera ono również informację o rozmiarze plików pakietów, jakie zostaną skopiowane ze źródła, co jest szczególnie istotne w przypadku dostępu przez sieć o małej prędkości. Podana jest informacja, ile miejsca zajmą w systemie plików instalowane pakiety. Po potwierdzeniu przez użytkownika, program przystępuje do kopiowania plików z wybranego przez niego źródła. Po pobraniu wszystkich plików, w tym przypadku poprzez serwer FTP, uruchamiane jest w tle **dpkg**. Dalej instalowanie przebiega praktycznie identycznie jak w przypadku użycia **dpkg**, czyli pliki pakietów są rozpakowywane, a następnie pakiety są konfigurowane.

## 2.4. Rekompilacja pakietów źródłowych

Przy pomocy APT można również dokonywać automatycznej rekompilacji pakietów źródłowych. Warunkiem pobrania i kompilacji pakietu źródłowego jest dopisanie w pliku `sources.list` URI wskazującego na lokalizację pakietów źródłowych (`deb-src`).

Aby pobrać i zrekompilować pakiet źródłowy należy wywołać `apt-get`:

```
# apt-get -b source nazwa_pakietu
```

Instalowany pakiet binarny będący wynikiem kompilacji może być zoptymalizowany pod kątem konkretnego systemu na którym jest instalowany, na przykład dla konkretnej architektury sprzętowej. Debian jest standardowo dostępny w wersji i386 na platformę Intel x86 i M68000 na platformę Motorola M68k. Jest tak między innymi dlatego, że rekompilacja całej dystrybucji (aktualnie ponad 4000 pakietów!) nie ma sensu i niewiele daje. Tym niemniej można sobie wyobrazić, że chce się zrekompilować konkretny pakiet, na przykład z optymalizacją dla procesorów Pentium. W takim przypadku można odpowiednio skonfigurować kompilator, lub nawet zainstalować odpowiedni pakiet Debiana, *pentium-builder*, i w trakcie pobierania pakietu przez APT nakazać mu automatyczną rekompilację.

Dzięki możliwości rekompilacji pakietów źródłowych można dokonywać złożonych uaktualnień dystrybucji, używając pakietów źródłowych z nowszych wersji Debiana.

## 2.5. APT i Dselect

Jak już wspomniano, docelowo system APT ma zastąpić `dselect`. Ponieważ jednak interfejsy graficzne użytkownika do APT nie są jeszcze gotowe, został on tymczasowo zintegrowany z `dselect` poprzez `apt-get`. W `dselect` jest dostępna dodatkowa metoda (`apt-get`) umożliwiająca otrzymywanie pakietów przez APT. Ta metoda była już dostępna w dystrybucji Debian/GNU 2.1, wymagała jednak ręcznej edycji pliku `sources.list`.

W przypadku dystrybucji 2.2 („Potato”) `dselect` udostępnia skrypt konfiguracyjny dla metody `apt-get`. Poniżej jest zaprezentowany przykładowy przebieg konfiguracji:

```
Access method 'apt'.
```

```
apt - APT Acquisition [file,http,ftp]
```

```
Set up a list of distribution source locations
```

```
Please give the base URL of the debian distribution.
```

```
URL [http://http.us.debian.org/debian]:
```

```
ftp://debian.mps.krakow.pl/mirror/debian
```

```
Please give the distribution tag...
```

```
Distribution [stable]: potato
```

```
Please give the components to get
```

```
Components [main contrib non-free]: main contrib
```

```
Get:1 ftp://debian.mps.krakow.pl potato/main Packages
```

```
Get:2 ftp://debian.mps.krakow.pl potato/main Release
```

```
Get:3 ftp://debian.mps.krakow.pl potato/contrib Packages
```

```
Get:4 ftp://debian.mps.krakow.pl potato/contrib Release
Reading Package Lists... Done
Building Dependency Tree... Done
Merging Available information... Done
Replacing available packages info, using /var/cache/apt/available.
Information about 4060 package(s) was updated.
```

Biorąc pod uwagę ogromne możliwości `apt-get`, jest to obecnie prawdopodobnie najlepsza metoda otrzymywania pakietów dla `dselect`.

## 2.6. Konfigurowanie APT

Konfiguracja całego systemu APT znajduje się pliku `/etc/apt.conf`. Korzystają z niej praktycznie wszystkie narzędzia z rodziny APT. W pliku znajdują się informacje o działaniu podstawowych funkcji APT, oraz jego poszczególnych interfejsów, takich jak interfejs do `dselect`, czy `dpkg`. Pełne omówienie składni tego pliku wykracza poza ramy artykułu, lecz jest ona dokładnie opisana w `apt.conf` (5). Konfiguracja w tym pliku może być uaktualniana ręcznie poprzez jego edycję, lub półautomatycznie poprzez program `apt-config`.

## 2.7. Dokumentacja APT

System APT ma wyczerpującą dokumentację, na którą składają się podręczniki oraz dokumentacja napisana w języku SGML, która jest dostępna w kilku formatach, na przykład HTML. Składa się ona z następujących części:

**cache** pełny opis implementacji `cache`'u APT'a, przechowującego informacje o dostępnych pakietach.

**design** ogólny opis architektury systemu APT.

**dpkg-tech** opis interfejsu pomiędzy APT a `dpkg`.

**files** dokument zawierający opis drzewa plików APT, w tym struktury plików konfiguracyjnych.

**guide** jest to podręcznik pozwalający na szybkie zapoznanie się z `apt-get`, jego konfiguracją i współpracą z `dselect`.

**method** opis interfejsu różnych metod dostępu do pakietów używanych przez APT.

**offline** wskazówki dotyczące używania APT na maszynie bez dostępu do sieci.

Oprócz wymienionej dokumentacji przydatne są dokumenty opisujące pakiety dodatkowe (na przykład `apt-zip`).

## 3. Alien

Program `alien` nie jest wprowadzić integralną częścią systemu pakietów, w takim sensie w jakim jest nią `dpkg`, `dselect` czy APT, pełni jednak bardzo ważną funkcję pomocniczą. `alien` służy do konwertowania pakietów pomiędzy różnymi formatami. Pierwotnie program miał służyć do dokonywania konwersji pomiędzy pakietami DEB i RPM, lecz w chwili obecnej wspiera wszystkie popularne formaty pakietów. Dzięki temu narzędziu możliwe jest instalowanie w systemie Debian/GNU pakietów przygotowanych dla dystrybucji RedHat, czy Slackware. Można również udostępniać pakiety DEB użytkownikom tych dystrybucji w ich formatach.

### 3.1. Funkcje i sposób użycia

Podstawową funkcją **alien** jest dokonywanie konwersji pakietów, domyślnie do formatu DEB. Konwersja może przebiegać w pełni automatycznie. W takim przypadku wystarczy wywołać program jedynie z nazwą pliku pakietu, na przykład w formacie RPM:

```
# alien gftp-1.13-4.i386.rpm
-- Examining gftp-1.13-4.i386.rpm
-- Unpacking gftp-1.13-4.i386.rpm
1108 blocks
-- Automatic package debianization
-- Building the package gftp_1.13-5_i386.deb
dh_testdir ... dh_builddeb
dpkg-deb: building package 'gftp' in '../gftp_1.13-5_i386.deb'.
Generation of gftp_1.13-5_i386.deb complete.
-- Successfully finished
```

Podobny efekt można uzyskać podając opcję **-d**. Z kolei dokonanie konwersji odwrotnej, to znaczy z formatu RPM do DEB, jest pokazane poniżej:

```
# alien -r vim_5.3-13.deb
-- Examining vim_5.3-13.deb
-- Unpacking vim_5.3-13.deb
-- Automatic spec file generation
-- Building the package /usr/src/redhat/RPMS/i386/vim-5.3-14.i386.rpm
Processing files: vim
Finding provides...
Finding requires...
Prereqs: /bin/sh
Wrote: /usr/src/redhat/RPMS/i386/vim-5.3-14.i386.rpm
Generation of /usr/src/redhat/RPMS/i386/vim-5.3-14.i386.rpm complete.
-- Successfully finished
```

Oprócz konwersji do formatu DEB (opcja **-d**) i formatu RPM (opcja **-r**), możliwa jest jeszcze konwersja do formatu **.tar.gz** używanego na przykład w dystrybucji Slackware. Czasami konwertowany pakiet może wymagać modyfikacji związanych z różnicami pomiędzy dystrybucjami. Jeżeli dokonuje się konwersji do formatu DEB, to przy pomocy opcji **--patch=nazwa\_pliku** można podać plik zawierający odpowiednią łatę. Można również jedynie rozpakować oryginalny pakiet (na przykład pakiet RPM) i ręcznie dokonać modyfikacji. W takim przypadku trzeba użyć opcji **-g**. **Alien** może od razu wywołać **dpkg** i zainstalować pakiet, wystarczy podać opcję **-i**. Nie jest to jednak polecane, jeżeli nie jest się pewnym tego, jak ostatecznie wygląda pakiet po konwersji.

Pomimo, że **alien** potrafi dobrze konwertować pakiety różnych dystrybucji, samo ich używanie nie musi być wcale takie łatwe. Jest ku temu kilka powodów. Po pierwsze dystrybucje mają różne systemy sprawdzania zależności. Pomimo, że **alien** dość dobrze radzi sobie z translacją tych zależności, może dojść do sytuacji, w której pakiet po konwersji nie będzie miał wszystkich niezbędnych informacji o zależnościach. Kolejną przeszkodą mogą być różnice w stosowanej w danej dystrybucji organizacji systemu plików. Teoretycznie, najważniejsze dystrybucje GNU/Linux są zgodne z FHS (*Filesystem Hierarchy Standard*). W praktyce standard dopuszcza w niektórych przypadkach pewną dowolność, a to może powodować problemy. Przykładowo, nie ma większych problemów przy przenoszeniu pakietów pomiędzy dystrybucjami Debian/GNU i RedHat, zakładając, że spełnione są wszystkie zależności (na przykład są

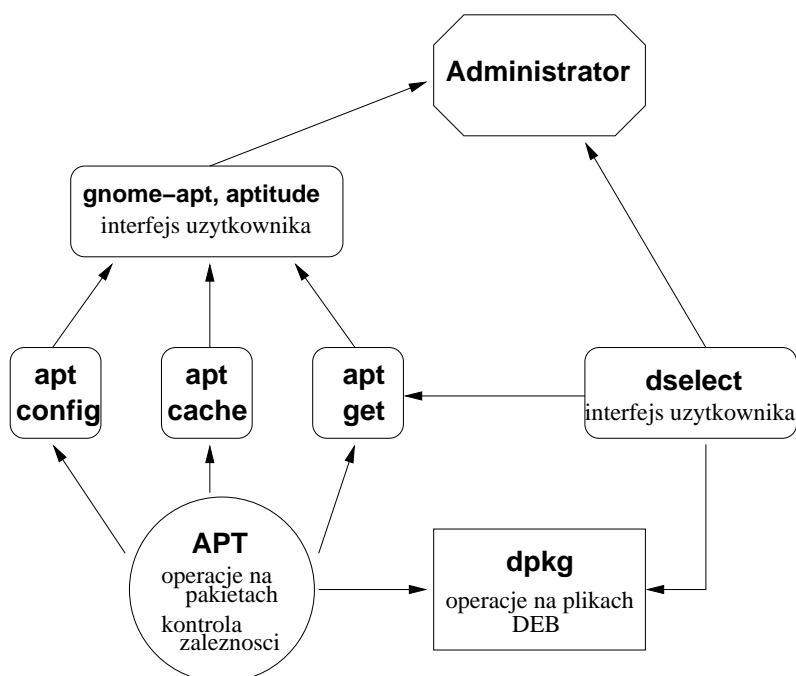


odpowiednie wersje bibliotek). Problemy pojawiają się natomiast z młodszymi dystrybucjami, których twórcy zdają się traktować FHS dość niefrasobliwie. Może się zdarzyć, że nawet w obrębie dystrybucji opartych o jeden format pakietów, na przykład dystrybucji RedHat, Suse, Caldera opartych o RPM, mogą mieć miejsce problemy związane z niezgodnościami w systemie plików.

Jak widać **alien** jest narzędziem przydatnym i prostym w obsłudze. Może być pomocny w sytuacjach, gdy nie jest dostępny pakiet binarny w żądanym formacie, a rekompilacja pakietu z kodem źródłowym jest trudna lub niemożliwa. Należy go jednak używać rozsądnie, pamiętając o wymienionych powyżej ograniczeniach w przenoszeniu pakietów.

## 4. Architektura systemu pakietów

Na zakończenie warto, w ramach podsumowania, przedstawić zarys architektury systemu pakietów w systemie Debian/GNU, przynajmniej na etapie wersji 2.2. Rysunek 1 przedstawia ogólny zarys tej architektury. Jak widać jest ona rozwinięciem tego, co było znane z wcześniejszych



Rysunek 1: Architektura systemu pakietów

wersji Debiana. System APT współistnieje z **dselect**, lecz stopniowo będzie go wypierał, wraz z pojawieniem się jego zaawansowanych interfejsów użytkownika. Jednym z podstawowych zalet tej architektury jest możliwość pełnej integracji systemu pakietów z siecią. Zaawansowany system sprawdzania zależności pozwala nie tylko na łatwe uaktualnianie pojedynczych pakietów, lecz również całej dystrybucji. Niewykluczone, że w przyszłości wraz z rozwojem systemu ta architektura zostanie uzupełniona o nowe elementy.

Należy przypomnieć, że pakiety dystrybucji Debian/GNU są zgodne z tzw. *Debian Policy*, czyli zbiorem ścisłych zasad dotyczących tworzenia, uaktualniania i rozwiązywania zależności pomiędzy pakietami. Polityka tworzenia pakietów dokładnie określa wspólne dla całej dystrybucji zasady tworzenia i uaktualniania plików konfiguracyjnych, dzielonych aplikacji i innych zasobów. Jej dokładne omówienie niestety wykracza poza ramy tego artykułu – zainteresowanych można odesłać do stron WWW Debian/GNU. Warto jednak zaznaczyć, że to właśnie

między innymi *Debian Policy* całość dystrybucji, która w przypadku wersji „Woody” składa się z około 5000 pakietów, jest niezwykle stabilna i ściśle zintegrowana.

## 5. Podsumowanie

Seria artykułów o systemie pakietów Debian/GNU była próbą scharakteryzowania jednego, prawdopodobnie najważniejszego, z elementów systemu Debian. Nie jest to bynajmniej próba opisu dystrybucji jako całości. Ze specyfiką Debian/GNU i jego systemu pakietów wiążą się przykładowo inne, nie opisane tutaj, zagadnienia:

- zaawansowany system Menu, pozwalający na automatyczną integrację z dowolnym *window managerem*,
- narzędzia do automatycznego konfigurowania pakietów,
- oprogramowanie wspierające tworzenie pakietów źródłowych, które można kompilować na dowolnej, wspieranej przez Debiana platformie,
- system wspierający alternatywne wersje pakietów (*/etc/alternatives*),
- zaawansowane narzędzia do konfiguracji i rekompilacji jądra systemu Linux,
- zcentralizowany system dokumentacji, dostępny lokalnie i przez WWW,
- zaawansowane narzędzia do obróbki dokumentów w językach opartych o SGML (DocBook, LinuxDoc, Html, DebianDoc itp.),
- wsparcie dla różnorodnych wersji programów (na przykład 3 serwery SMTP, kilka serwerów HTTP, itd.)
- i wiele, wiele innych elementów.

Przykłady można mnożyć, a ich pełniejsze omówienie można znaleźć w dokumentacji dostępnej w systemie, lub w książkach poświęconych Debian/GNU. Można mieć nadzieję, że omówione zagadnienia przedstawione w tym i w poprzednich artykułach okażą się pomocne dla administratorów i użytkowników Debian/GNU, a także przybliżą ten system tym, którzy go nie znają.