Debian Entwickler-Referenz

Autoren der Entwickler-Referenz, Andreas Barth, Adam Di Carlo, Raphaël Hertzog, Lucas Nussbaum, Christian Schwarz und Ian Jackson

25. Juni 2012

Debian Entwickler-Referenz
by Autoren der Entwickler-Referenz, Andreas Barth, Adam Di Carlo, Raphaël Hertzog, Lucas Nussbaum, Christian Schwarz und Ian Jackson

Published 2012-06-25

Wenn Sie diese Referenz ausdrucken möchten, sollten Sie die PDF-Version verwenden. Diese Seite ist auch auf Englisch und Französisch und Japanisch verfügbar. Die deutsche Übersetzung wurde 2011 von Chris Leick <c.leick@vollbio.de> verfasst.

Inhaltsverzeichnis

Kapitel 1

Geltungsbereich dieses Dokuments

Dieses Dokument soll Debian-Entwicklern einen Überblick über die empfohlenen Prozeduren und die verfügbaren Ressourcen geben.

Die hier besprochenen Prozeduren umfassen die Bewerbung als Paketbetreuer (Kapitel 2), die Erstellung neuer Pakete (Abschnitt 5.1), das Hochladen von Paketen (Abschnitt 5.6), die Handhabung von Fehlerberichten (Abschnitt 5.8), das Verschieben, Entfernen oder Verwaisen von Paketen (Abschnitt 5.9), das Portieren von Paketen (Abschnitt 5.10) und wie und wann Pakete anderer Paketbetreuer vorläufig veröffentlicht werden (Abschnitt 5.11).

Die in dieser Referenz besprochenen Ressourcen umfassen die Mailinglisten (Abschnitt 4.1) und Server (Abschnitt 4.4), eine Erörterung der Struktur des Debian-Archivs (Abschnitt 4.6), eine Erklärung der unterschiedlichen Server, die das Hochladen von Paketen akzeptieren (Abschnitt 5.6.1) und eine Erörterung von Mitteln, die Paketbetreuern dabei helfen, die Qualität ihrer Pakete zu gewährleisten (Anhang A).

Es sollte klar sein, dass diese Referenz nicht die technischen Einzelheiten von Debian-Paketen erörtert oder wie sie erstellt werden. Ebenso wenig wird diese Referenz die Standards einzeln aufführen, die Debian-Software erfüllen muss. All diese Informationen finden Sie im Debian Policy Manual.

Weiterhin ist dieses Dokument *nicht ein Ausdruck einer formalen Richtlinie*. Es enthält Dokumentation für das Debian-System und allgemein vereinbarte gute fachliche Gebräuche. Daher ist es nicht das, was normalerweise als ein »normgebendes« Dokument bezeichnet wird.

Kapitel 2

Bewerbung als Betreuer

2.1 Erste Schritte

Also, Sie haben all die Dokumentationen gelesen, Sie sind dieAnleitung für zukünftige Debian-Betreuer durchgegangen, verstehen alles, was im Beispielpaket `hello` vorkommt und es geht Ihnen darum, Ihre Lieblings-Software zu debianisieren. Möchten Sie wirklich ein Debian-Entwickler werden, damit Ihre Arbeit in das Projekt aufgenommen wird?

Falls Sie das nicht bereits getan haben, schreiben Sie sich zuerst in debian-devel@lists.debian.org ein. Senden Sie das Wort `subscribe` im Betreff einer E-Mail an debian-devel-REQUEST@lists.debian.org. Falls es Probleme gibt, wenden Sie sich unter listmaster@lists.debian.org an den Administrator der Liste. Weitere Informationen über verfügbare Listen finden Sie unter Abschnitt 4.1. debian-devel-announce@lists.debian.org ist eine Pflichtlektüre für jeden, der die Entwicklung von Debian verfolgen möchte.

Sie sollten sich einschreiben und ein wenig mitlesen ohne selbst Beiträge zu verfassen, bevor Sie irgend etwas codieren. Teilen Sie mit an was Sie arbeiten möchten, um doppelten Aufwand zu vermeiden.

Auch debian-mentors@lists.debian.org ist ein sinnvolles Abonnement. Lesen Sie Abschnitt 2.2 für Details. Auch der IRC-Kanal `#debian` kann hilfreich sein. Siehe Abschnitt 4.2.

Wenn Sie wissen ,wie Sie zu Debian GNU/Linux beitragen möchten, sollten Sie Kontakt mit existierenden Debian-Betreuern aufnehmen, die an ähnlichen Aufgaben arbeiten. Auf diesem Weg können Sie von erfahrenen Entwicklern lernen. Falls Sie zum Beispiel daran interessiert sind, in Debian existierende Software zu paketieren, sollten Sie versuchen einen Sponsor zu finden. Ein Sponsor wird mit Ihnen zusammen an Ihrem Paket arbeiten und es in das Debian-Archiv hochladen, sobald er mit Ihrer Paketierung zufrieden ist. Sie können Sponsoren auf der Mailingliste debian-mentors@lists.debian.org finden, indem Sie dort Ihr Paket und sich selbst beschreiben und nach einem Sponsor fragen (lesen Sie Abschnitt 7.5.1 und `http://wiki.debian.org/DebianMentorsFaq`, um weitere Informationen über Sponsoring zu erhalten). Wenn Sie andererseits daran interessiert sind, Debian auf alternative Architekturen oder Kernel zu portieren, können Sie spezielle Mailinglisten zur Portierung abonnieren und dort nachfragen, wie der Einstieg gelingt. Abschließend, falls Sie daran interessiert sind, an der Dokumentation oder Qualitätssicherung (QS) zu arbeiten, können Sie sich den Betreuern anschließen, die an diesen Aufgaben arbeiten und Patches sowie Verbesserungen einsenden.

Eine Schwierigkeit könnte ein zu generischer lokaler Teil in Ihrer Mailadresse sein. Begriffe wie »mail«, »admin«, »root« und »master« sollten vermieden werden. Bitte lesen Sie `http://www.debian.org/MailingLists/` für Details.

2.2 Debian-Mentoren und -Sponsoren

Die Mailingliste debian-mentors@lists.debian.org wurde für Einsteiger unter den Betreuern eingerichtet, die sich um Hilfe beim Paketieren Ihrer ersten Pakete und anderen entwicklerbezogenen Themen bemühen. Jeder neue Entwickler ist eingeladen, sich auf dieser Liste einzuschreiben (siehe Abschnitt 4.1 für Details).

Diejenigen, die persönliche Hilfe bevorzugen (z.B. mittels privater E-Mail), sollten auch an diese Liste schreiben und ein erfahrener Entwickler wird freiwillig helfen.

Falls Sie darüberhinaus einige Paket haben, die bereit für die Aufnahme in Debian sind, aber darauf warten, dass Sie Ihre Bewerbung als neuer Betreuer durchlaufen, könnten Sie einen Sponsor suchen, der Ihr Paket für Sie hochlädt. Sponsoren sind offizielle Debian-Entwickler und bereit, Pakete für Sie zu beurteilen und hochzuladen. Bitte lesen Sie zuerst die FAQ für Debian-Mentoren unter `http://wiki.debian.org/DebianMentorsFaq`.

Falls Sie ein Mentor und/oder Sponsor werden möchten, sind weitere Informationen in Abschnitt 7.5 verfügbar.

2.3 Als Debian-Entwickler registrieren

Bevor Sie sich entscheiden, sich bei Debian GNU/Linux zu registrieren, werden Sie alle in Debians New-Maintainer-Ecke verfügbaren Informationen lesen müssen. Dort wird im Detail beschrieben, welche Vorbereitungen Sie treffen müssen, bevor Sie sich registrieren können, um Debian-Entwickler zu werden. Zum Beispiel müssen Sie, bevor Sie sich bewerben, den Debian-Gesellschaftsvertrag lesen. Die Registrierung als Debian-Entwickler bedeutet, dass Sie ihm zustimmen und versprechen, den Debian-Gesellschaftsvertrag zu unterstützen. Es ist sehr wichtig, dass Betreuer mit den grundsätzlichen Ideen hinter Debian GNU/Linux einverstanden sind. Es wäre außerdem empfehlenswert, das GNU-Manifest zu lesen.

Der Aufnahmeprozess für Entwickler werden Ihre Identität, Absichten und technischen Fähigkeiten geprüft werden. Als die Anzahl der Leute, die an Debian GNU/Linux arbeiteten auf über 1000 wuchs und das System an mehreren sehr wichtigen Stellen lief, war Vorsicht geboten, um nicht kompromittiert zu werden. Daher müssen neue Betreuer überprüft werden, bevor Sie Konten auf den Servern erhalten und Pakete hochladen dürfen.

Bevor Sie sich tatsächlich registrieren, sollten Sie zeigen, dass Sie kompetent arbeiten und gute Beiträge leisten. Sie zeigen dies, indem Sie Patches an die Debian-Fehlerdatenbank senden und ein Paket haben, das ein existierender Debian-Entwickler für eine Zeit lang sponsort. Außerdem wird erwartet, dass Mitwirkende sich für das ganze Projekt interessieren und nicht nur ihre eigenen Pakete betreuen. Falls Sie anderen Betreuern helfen können, weitere Informationen zu einem Fehler oder sogar einen Patch bereitzustellen, dann tun Sie dies!

Für die Registrierung ist es erforderlich, dass Sie mit der Debian-Philosophie und der technischen Dokumentation vertraut sind. Weiterhin benötigen Sie einen GnuPG-Schlüssel, der von einem existierenden Debian-Betreuer signiert wurde. Falls Ihr GnuPG-Schlüssel noch nicht signiert wurde, sollten Sie versuchen einen Debian-Entwickler persönlich zu treffen, damit Ihr Schlüssel signiert wird. Es gibt eine GnuPG Key Signing Coordination page, die Ihnen helfen sollte, einen Debian-Entwickler in Ihrer Nähe zu finden. (Falls es in Ihrer Nähe keinen Debian-Entwickler gibt, können als absolute Ausnahme fallspezifisch alternative Identitätsprüfungen erlaubt werden. Lesen Sie die Texte zur Identitätsprüfung, um weitere Informationen zu erhalten.

Falls Sie noch keinen OpenPGP-Schlüssel haben, generieren Sie sich einen. Jeder Entwickler benötigt einen OpenPGP-Schlüssel, um das Hochladen von Paketen zu signieren und zu prüfen. Sie sollten das Handbuch der Software lesen, die Sie benutzen, da es wichtige Informationen enthält, was für die Sicherheit kritisch ist. Es sind viel mehr Sicherheitslücken auf menschliche Fehler zurückzuführen, als auf Softwarefehler oder leistungsstarke Spionagetechniken. Lesen Sie Abschnitt 3.2.2, um weitere Informationen über die Verwaltung öffentlicher Schlüssel zu erhalten.

Debian benutzt den `GNU Privacy Guard` (Paket `gnupg` Version 1 oder besser) als grundlegenden Standard. Sie können auch einige andere Implementierungen von OpenPGP benutzen. Beachten Sie, dass OpenPGP ein offener Standard ist, der auf RFC 2440 basiert.

Für die Debian-Entwicklung benötigen Sie einen Schlüssel der Version 4. Ihr Schlüssel muss mindestens 1024 Bit lang sein. Es gibt keinen Grund, einen kürzeren Schlüssel zu verwenden und dies wäre auch wesentlich unsicherer.[1]

Falls Ihr öffentlicher Schlüssel nicht auf einem öffentlichen Server wie `subkeys.pgp.net` liegt, lesen Sie bitte NM Schritt 2: Identitätsprüfung. Dieses Dokument enthält Anweisungen, wie Sie Ihren Schlüssel auf öffentliche Schlüsselserver ablegen. Die »New Maintainer Group« wird Ihren öffentlichen Schlüssel auf den Servern ablegen, wenn er nicht bereits dort ist.

Einige Länder schränken den Gebrauch kryptografischer Software durch Ihre Bürger ein. Dies muss jedoch die Aktivitäten als Debian-Paketbetreuer nicht behindern, da es vollkommen legal sein kann, kryptografische Produkte für die Authentifizierung anstatt für Verschlüsselungszwecke zu verwenden. Wenn Sie in einem Land leben, in dem Kryptografie sogar für Authentifizierung verboten ist, so kontaktieren Sie bitte Debian, so dass besondere

[1] Schlüssel der Version 4 folgen dem in RFC 2440 definierten OpenPGP-Standard. Bei Verwendung von GnuPG wurden immer Version-4-Schlüssel erstellt. PGP-Versionen seit 5.x könnten außerdem Schlüssel der Version 4 erstellen, die andere Wahl waren PGP 2.6.x-kompatible Schlüssel der Version 3 (auch altes RSA durch PGP genannt).

(Primär-) Schlüssel der Version 4 können entweder die RSA- oder DSA-Algorithmen benutzen. Das hat dies nichts mit der Frage von GnuPG zu tun, welche Schlüsselart Sie möchten: (1) DSA und Elgamal, (2) DSA (nur signieren), (5) RSA (nur signieren). Falls Sie keine besonderen Anforderungen haben, nehmen Sie ruhig die Voreinstellung.

Die einfachste Möglichkeit festzustellen, ob ein existierender Schlüssel Version 3 oder 4 (oder 2) hat, besteht darin auf den Fingerabdruck zu untersuchen: Fingerabdrücke der Version 4 sind der SHA-1-Hash von einigem Schlüsselmaterial. Daher besteht er aus 40 hexadezimale Ziffern, die üblicherweise in Viererblöcken gruppiert sind. Fingerabdrücke älterer Schlüsselformatversionen benutzten MD5 und werden generell als Blöcke zweier hexadezimaler Ziffern angezeigt. Falls Ihr Fingerabdruck zum Beispiel so aussieht wie `5B00 C96D 5D54 AEE1 206B A F84 DE7A AF6E 94C0 9C7F`, dann ist es ein Schlüssel der Version 4.

Eine andere Möglichkeit besteht darin, den Schlüssel über eine Pipe an **pgpdump** zu leiten, das dann etwas wie »Public Key Packet - Ver 4.« ausgibt.

Beachten Sie außerdem, dass Ihr Schlüssel selbstsigniert sein muss (d.h. er muss für alle eigenen Benutzer-IDs signiert werden. Dies verhindert eine Manipulation von Benutzer-IDs). Sämtliche modernen OpenPGP-Programme erledigen dies automatisch. Falls Sie aber einen älteren Schlüssel haben, müssen Sie diese Signaturen manuell hinzufügen.

Vereinbarungen getroffen werden können.

Um sich als neuer Betreuer zu bewerben, benötigen Sie die Fürsprache eines existierenden Debian-Entwicklers, der Ihre Bewerbung unterstützt (einen `advocate`). Nachdem Sie eine Weile zu Debian beigetragen haben und Sie sich als ein registrierter Entwickler bewerben, benötigen Sie die Fürsprache eines existierenden Entwicklers, mit dem Sie während der letzten Monate zusammengearbeitet haben. Dieser muss sein Vertrauen kundtun, dass Sie erfolgreich zu Debian beitragen können.

Wenn Sie einen Fürsprecher gefunden haben, Ihr GnuPG-Schlüssel signiert ist und Sie bereits eine Weile zu Debian beigetragen haben, sind Sie bereit, sich zu bewerben. Sie können sich einfach auf der Bewerbungsseite für neue Betreuer registrieren. Nachdem Sie unterschrieben haben, muss Ihr Befürworter Ihre Bewerbung bestätigen. Wenn Ihr Befürworter diesen Schritt abgeschlossen hat, wird Ihnen ein Bewerbungsbetreuer zugewiesen, der mit Ihnen die nötigen Schritte des Prozesses für neue Betreuer durchläuft. Sie können jederzeit den Status auf der Bewerberstatusanzeige prüfen.

Für weitere Details besuchen Sie bitte die New-Maintainer-Ecke auf der Debian-Website. Stellen Sie sicher, dass sie mit den nötigen Schritten des Prozesses für neue Betreuer vertraut sind, bevor Sie sich tatsächlich bewerben. Wenn Sie gut vorbereitet sind, können Sie nachher viel Zeit sparen.

Kapitel 3

Pflichten von Debian-Entwicklern

3.1 Pflichten von Paketbetreuern

Als Paketbetreuer sollen Sie Pakete hoher Qualität bereitstellen, die gut in das System integriert sind und die den Debian-Richtlinien folgen.

3.1.1 Auf die nächste `stable`-Veröffentlichung hinarbeiten

Es reicht nicht aus, Pakete hoher Qualität in `unstable` bereitzustellen, die meisten Anwender werden nur von Ihren Paketen profitieren, wenn Sie Teil der nächsten `stable`-Veröffentlichung sind. Es wird daher von Ihnen erwartet, dass Sie mit dem Release-Team zusammenarbeiten, um sicherzustellen, dass Ihre Pakete dort berücksichtigt werden.

Konkreter ausgedrückt, sollten Sie überwachen, ob Ihre Pakete nach `testing` (siehe Abschnitt 5.13) wandern. Wenn die Migration nicht nach der Testperiode stattfindet, sollten Sie analysieren, warum und darauf hinarbeiten, dies zu beheben. Es könnte heißen, das Sie Ihr Paket reparieren müssen (im Fall veröffentlichungskritischer Fehler oder wenn das Bauen auf einigen Architekturen fehlschlägt), aber es kann auch heißen, dass andere Pakete aktualisiert (oder repariert oder aus `testing` entfernt) werden müssen, um bei einem Übergang zu helfen, in den Ihr Paket aufgrund von Abhängigkeiten verstrickt ist. Das Release-Team könnte Ihnen einige Informationen liefern, was derzeit einen gegebenen Übergang blockiert, falls Sie das nicht erkennen können.

3.1.2 Pakete in `stable` betreuen

Die meiste Arbeit von Paketbetreuern geht in das Bereitstellen aktualisierter Paketversionen in `unstable`, aber ihre Aufgabe bedingt auch, sich um Pakete in der aktuellen Veröffentlichung von `stable` zu kümmern.

Obwohl von Änderungen in `stable` abgeraten wird, sind diese dennoch möglich. Immer wenn ein Sicherheitsproblem gemeldet wird, sollten Sie mit dem Sicherheits-Team zusammenarbeiten, um eine reparierte Version bereitzustellen (siehe Abschnitt 5.8.5). Wenn Fehler des Schweregrads »important« (oder höher) gemeldet werden, sollten Sie in Betracht ziehen, eine gezielte Fehlerbehebung zur Verfügung zu stellen. Sie können das `stable`-Release-Team fragen, ob es eine solche Aktualisierung akzeptieren würde und dann einen `stable`-Upload vorbereiten (siehe Abschnitt 5.5.1).

3.1.3 Verwalten veröffentlichungskritischer Fehler

Sie sollten Fehlerberichte zu Ihren Paketen generell so erledigen, wie es in Abschnitt 5.8 beschrieben wird. Es gibt jedoch eine spezielle Fehlerkategorie, auf die Sie besonders Acht geben sollten – sogenannte veröffentlichungskritische Fehler (release critical bugs/RC-Fehler). Alle Fehlerberichte, mit den Schweregrad `critical`, `grave` oder `serious` machen das Paket ungeeignet für eine Aufnahme in die nächste `stable`-Veröffentlichung. Sie können daher die Debian-Veröffentlichung verzögern (wenn sie ein Paket in `testing` beeinflussen) oder Migrationen nach `testing` blockieren (wenn sie nur ein Paket in `unstable` beeinflussen). Im schlimmsten Fall führen Sie zum Entfernen des Pakets. Daher müssen diese Fehler so schnell wie möglich behoben werden.

Falls Sie aus irgend einem Grund nicht in der Lage sind, den Fehler in einem Paket innerhalb von zwei Wochen zu beheben (zum Beispiel aus Termingründen oder weil er schwer zu beheben ist), sollten Sie es klar im Fehlerbericht vermerken und den Fahler mit `help` kennzeichnen, um Freiwillige einzuladen, sich einzuschalten. Seien Sie sich bewusst, dass veröffentlichungskritische Fehler oft das Ziel von Uploads durch Nicht-Betreuer (»Non-Maintainer Uploads«, siehe Abschnitt 5.11) sind, da sie die Migration vieler Pakete nach `testing` blockieren können.

Mangel an Aufmerksamkeit gegenüber veröffentlichungskritischen Fehler wird vom Release-Team oft als Zeichen interpretiert, dass der Betreuer verschwunden ist, ohne sein Paket ordentlich zu verwaisen. Das MIA-Team könnte außerdem eingeschaltet werden, was dazu führen könnte, dass Ihre Pakete verwaist werden (siehe Abschnitt 7.4).

3.1.4 Abstimmung mit Originalautoren

Ein Großteil Ihrer Arbeit als Debian-Betreuer wird darin bestehen, mit den Autoren des Programms Kontakt zu halten. Manchmal melden Debian-Anwender Fehler an die Debian-Fehlerdatenbank, die nicht Debian-spezifisch sind. Sie müssen diese Fehler an die Programmautoren weiterleiten, so dass die Programmautoren sie in einer zukünftigen Veröffentlichung beheben können.

Obwohl es nicht Ihre Arbeit ist, nicht Debian-spezifische Fehler zu beheben, können Sie dies dennoch tun, wenn Sie dazu in der Lage sind. Wenn Sie solche Fehler beheben, stellen Sie sicher, dass Sie ihre Korrekturen auch an die ursprünglichen Betreuer weiterleiten. Debian-Anwender und -Entwickler werden manchmal Patche schicken, um Fehler im Ursprungsprogramm zu beheben – Sie sollten diese Patche auswerten und an die ursprünglichen Autoren weiterleiten.

Falls Sie die ursprünglichen Quellen verändern müssen, um ein den Richtlinien entsprechendes Paket zu erstellen, dann sollten Sie freundlich eine Korrektur vorschlagen, die dort eingefügt werden kann, so dass Sie die Quellen bei der nächsten Version der Urprungsautoren nicht ändern müssen. Ganz gleich, welche Änderungen sie möchten – versuchen Sie ein Verzweigen von den ursprünglichen Quellen zu vermeiden.

Wenn Sie der Ansicht sind, dass die ursprünglichen Entwickler Debian oder der Gemeinschaft rund um freie Software ablehnend gegenüber stehen, könnten Sie es sich nochmal überlegen, ob Sie die Software in Debian einbringen möchten. Manchmal sind es die gesellschaftlichen Kosten höher, als der Gewinn, den die Software mitbringt.

3.2 Verwaltungspflichten

Ein Projekt der Größe von Debian beruht auf einer Verwaltungsinfrastruktur, um über alles den Überblick zu behalten. Als Projektmitglied haben Sie einige Pflichten, um sicherzustellen, dass alles glatt läuft.

3.2.1 Verwaltung Ihrer Debian-Informationen

Es gibt unter `https://db.debian.org/` eine LDAP-Datenbank, die Informationen über Debian-Entwickler enthält. Sie sollten Ihre Informationen dort eintragen und bei Änderungen aktualisieren. Stellen Sie insbesondere sicher, dass Ihre Weiterleitungsadresse für Ihre debian.org-E-Mails immer aktuell ist. Das gilt auch für die Adresse, zu der Ihre »debian-private«-E-Mails versendet werden, wenn Sie sich auch dort angemeldet haben.

Um weitere Informationen über die Datenbank zu erhalten, lesen Sie bitte Abschnitt 4.5.

3.2.2 Verwalten Ihres öffentlichen Schlüssels

Gehen Sie sehr vorsichtig mit Ihren privaten Schlüsseln um. Legen Sie sie nicht auf öffentlichen Servern oder Maschinen mit mehreren Benutzern ab, wie den Debian-Servern (siehe Abschnitt 4.4). Sichern Sie Ihre Schlüssel. Behalten Sie eine Kopie offline. Lesen Sie die Dokumentation, die Ihrer Software beiliegt. Lesen Sie die PGP-FAQ.

Sie müssen nicht nur dafür sorgen, dass Ihr Schlüssel sicher vor Diebstahl ist, sondern auch, dass er nicht verloren geht. Generieren Sie Ihr Widerrufs-Zertifikat und erstellen Sie eine Kopie davon (am besten in Papierform). Dies wird benötigt, wenn Sie Ihren Schlüssel verloren haben.

Falls Sie Ihrem öffentlichen Schlüssel Signaturen oder Benutzerkennungen hinzufügen, können Sie den Debian-Schlüsselring aktualisieren, indem Sie Ihren Schlüssel an den Schlüsselserver unter `keyring.debian.org` senden.

Wenn Sie einen komplett neuen Schlüssel hinzufügen oder einen alten entfernen möchten, benötigen Sie den durch einen anderen Entwickler neu signierten Schlüssel. Falls der alte Schlüssel kompromittiert oder ungültig ist, müssen Sie außerdem das Widerrufs-Zertifikat hinzufügen. Falls es keinen vernünftigen Grund für einen neuen Schlüssel gibt, können die Betreuer des Schlüsselrings den neuen Schlüssel ablehnen. Details finden Sie unter `http://keyring.debian.org/replacing_keys.html`.

Es werden die gleichen Schlüssel-Extrahierungsroutinen angewandt, wie sie unter Abschnitt 2.3 besprochen wurden.

Eine weiterreichende Erörterung der Debian-Schlüsselverwaltung können Sie in der Dokumentation des Pakets `debian-keyring` finden.

3.2.3 Abstimmungen

Obwohl Debian nicht wirklich demokratisch ist, werden demokratische Prozesse benutzt, um das Führungspersonal auszuwählen und generellen Entschlüssen zuzustimmen. Diese Prozeduren sind durch die Debian-Verfassung definiert.

Im Gegensatz zur jährlichen Wahl des Projektleiters werden keine regelmäßigen Abstimmungen abgehalten und wenn doch, dann nicht einfach so. Jeder Vorschlag wird zuerst auf der Mailingliste debian-vote@lists.debian.org besprochen und benötigt mehrere Befürworter, bevor der Projektsekretär die Abstimmungsprozedur in Gang setzt.

Sie müssen vor der Abstimmung nicht alle Diskussionen verfolgen, da der Sekretär mehrere Aufrufe zur Abstimmung auf debian-devel-announce@lists.debian.org herausgibt (Und von allen Entwicklern wird erwartet, das sie diese Liste abonniert haben). Demokratie funktioniert nicht richtig, wenn die Leute nicht an Abstimmungen teilnehmen, daher wird allen Entwicklern empfohlen abzustimmen. Die Abstimmung wird mittels GPG-signierte/verschlüsselte E-Mails durchgeführt.

Die Liste aller(früheren und aktuellen) Vorschläge ist auf der Seite Debian-Abstimmungs-Informationen verfügbar, ebenso die Informationen, wie Vorschläge gemacht und unterstützt werden und darüber abgestimmt wird.

3.2.4 Elegant Urlaub machen

Es ist bei Entwicklern üblich, dass es Zeiten gibt, in denen sie abwesend sind, entweder wegen geplanten Ferien oder einfach, weil sie unter anderer Arbeit begraben sind. Am wichtigsten ist, dass Sie daran denken, andere Entwickler über Ihren Urlaub zu informieren, damit diese bei Problemen mit Ihren Paketen oder anderweitigen Pflichten im Projekt die erforderlichen Maßnahmen ergreifen können.

Üblicherweise bedeutet dies, dass andere Entwickler ein NMU (siehe Abschnitt 5.11) Ihres Pakets durchführen dürfen, weil ein großes Problem (veröffentlichungskritischer Fehler, Sicherheitsaktualisierung etc.) auftritt während Sie in Ferien sind. Manchmal ist dies etwas weniger kritisches, aber es ist trotzdem angemessen, andere wissen zu lassen, dass Sie nicht verfügbar sind.

Um andere Entwickler zu informieren, gibt es zwei Dinge, die Sie tun sollten. Zuerst senden Sie eine E-Mail an debian-private@lists.debian.org bei der Sie [VAC] vor den Betreff Ihrer Nachricht schreiben[1] und die Dauer angeben, wie lange Sie Urlaub machen. Sie können außerdem einige besondere Anweisungen geben, was beim Auftreten von Fehlern zu tun ist.

Das andere, was Sie tun sollten, ist, sich selbst in der LDAP-Entwicklerdatenbank von Debian als abwesend zu kennzeichnen (auf diese Information können nur Debian-Entwickler zugreifen). Vergessen Sie nicht, die Urlaubskennzeichnung zu entfernen, wenn Sie wieder zurück sind.

Idealerweise sollten Sie sich auf den GPG-Koordinierungsseiten anmelden, wenn Sie Urlaub buchen und prüfen, ob es dort jemanden gibt, der eine Signatur benötigt. Dies ist besonders dann wichtig, wenn Leute an exotische Orte fahren, an denen es noch keine Entwickler gibt, aber Leute, die an einer Bewerbung interessiert sind.

3.2.5 Sich zurückziehen

Falls Sie das Debian-Projekt verlassen wollen, sollten Sie die folgenden Schritte einhalten:

1. Verwaisen Sie all Ihre Pakete, wie in Abschnitt 5.9.4 beschrieben.

2. Senden Sie eine GPG-signierte E-Mail mit den Gründen, weshalb Sie das Projekt verlassen, an debian-private@lists.debian.org.

3. Benachrichtigen Sie die Debian-Schlüsselverwalter über Ihren Weggang, indem Sie ein Ticket in Debian-RT öffnen. Dies geschieht durch Senden einer E-Mail an keyring@rt.debian.org mit den Wörtern »Debian RT« in der Betreffzeile (Groß- und Kleinschreibung spielt keine Rolle).

Es ist wichtig, obigem Prozess zu folgen, da die Suche nach inaktiven Entwicklern und das Verwaisen ihrer Pakete spürbar Zeit und Mühe kostet.

3.2.6 Nach dem Ausscheiden zurückkehren

Das Konto eines zurückgetretenen Entwicklers ist als »emeritus« markiert, wenn dem Prozess in Abschnitt 3.2.5 gefolgt wird und ansonsten als »disabled«. Zurückgetretene Entwickler mit einem »emeritus«-Konto können Ihr Konto wie folgt neu aktivieren:

[1] Dies geschieht so, dass die Nachricht einfach von Leuten gefiltert werden kann, die keine Urlaubsbenachrichtigungen lesen möchten.

- Kontaktieren Sie da-manager@debian.org.
- Durchlaufen Sie den verkürzten NM-Prozess (um sicherzustellen, dass der zurückgekehrte Entwickler noch immer die wichtigen Teile von P&P – »Philosophie und Prozeduren« und T&S – »Aufgaben und Fertigkeiten« kennt).
- Beweisen Sie, dass Sie immer noch den mit dem Konto verbundenen GPG-Schlüssel kontrollieren oder stellen Sie den Nachweis Ihrer Identität für einen neuen GPG-Schlüssel mit mindestens zwei Signaturen anderer Entwickler bereit.

Zurückgetretene Entwickler mit einem »disabled«-Konto müssen den NM-Prozess erneut durchlaufen.

Kapitel 4

Ressourcen für Debian-Entwickler

In diesem Kapitel werden Sie einen kurzen Leitfaden der Debian-Mailinglisten finden, der Debian-Maschinen, die Ihnen als Entwickler zur Verfügung stehen, und all den anderen Ressourcen, die verfügbar sind, um Ihnen bei Ihrer Arbeit als Betreuer zu helfen.

4.1 Mailinglisten

Viele Unterhaltungen zwischen Debian-Entwicklern (und Anwendern) werden durch ein weites Feld von Mailinglisten verwaltet, die auf `lists.debian.org` untergebracht sind. Mehr darüber, wie Sie sie abonnieren oder abbestellen, wie sie Artikel abschicken, wo alte Artikel gefunden werden und wie Sie sie suchen, wie Sie Listenbetreuer kontaktieren und verschiedene sonstige Informationen über Mailinglisten finden, können Sie unter `http://www.debian.org/MailingLists/` lesen. Dieser Abschnitt wird nur die Gesichtspunkte der Mailinglisten aufdecken, die von besonderem Interesse für Entwickler sind.

4.1.1 Grundregeln für die Benutzung

Wenn Sie auf Nachrichten auf der Mailingliste antworten, senden Sie bitte keine Kopie (`CC`) an den ursprünglichen Verfasser, außer, wenn dieser explizit darum bittet. Jeder, der an eine Mailingliste schreibt, sollte sie lesen, um die Antworten zu sehen.

Kreuzversand (die gleiche Nachricht an mehrere Listen senden) ist unerwünscht. Kürzen Sie, wie immer im Netz, die Zitate von Artikeln, auf die Sie antworten. Halten Sie sich bitte an die allgemeinen Gepflogenheiten beim Versand von Nachrichten.

Bitte lesen Sie den Leitfaden, um weitere Informationen zu erhalten. Die Debian Community Guidelines sind ebenfalls Wert, gelesen zu werden.

4.1.2 Haupt-Entwickler-Mailinglisten

Die Haupt-Debian-Mailinglisten, die Entwickler nutzen sollten, sind:

- debian-devel-announce@lists.debian.org wird benutzt, um Entwicklern wichtige Dinge anzukündigen. Es wird von allen Entwicklern erwartet, dass sie diese Liste abonnieren.
- debian-devel@lists.debian.org wird benutzt, um über verschiedene entwicklungsbezogene technische Themen zu reden.
- debian-policy@lists.debian.org wird benutzt, um über die Debian-Richtlinien zu diskutieren und darüber abzustimmen.
- debian-project@lists.debian.org wird benutzt, um über verschiedene entwicklungsbezogene, nicht technische Themen zu reden.

Es sind weitere Mailinglisten für unterschiedliche Spezialthemen verfügbar. Eine Liste finden Sie unter `http://lists.debian.org/`.

4.1.3 Spezielle Listen

debian-private@lists.debian.org ist eine besondere Mailingliste für private Unterhaltungen zwischen Debian-Entwicklern. Das heißt, sie sollte benutzt werden, um über Dinge zu reden, die aus irgend einem Grund nicht veröffentlicht werden sollen. Eigentlich ist es eine Liste mit geringem Umfang und Benutzer werden angehalten debian-private@lists.debian.org nicht zu benutzen, so lange es nicht wirklich nötig ist. Leiten Sie außerdem *keine* E-Mail von dieser Liste an jemanden weiter. Es sind aus naheliegenden Gründen keine Archive dieser Liste im Web verfügbar, aber Sie können sie sehen, indem Sie Ihr Shell-Konto auf `master.debian.org` benutzen und in das Verzeichnis `~debian/archive/debian-private/` schauen.

debian-email@lists.debian.org ist eine besondere Mailingliste, die als Wundertüte für Debian-bezogene Korrespondenz, wie den Kontakt zu ursprünglichen Autoren über Lizenzen, Fehler, etc. oder Diskussionen über das Projekt mit anderen benutzt wird, wobei es nützlich sein könnte, dass die Diskussion irgendwo archiviert wird.

4.1.4 Antrag auf neue entwicklungsbezogene Listen

Bevor Sie eine Mailingliste beantragen, die sich auf die Entwicklung eines Pakets (oder eine kleine Gruppe verwandter Pakete) bezieht, bedenken Sie bitte, dass es angebrachter ist, wenn Sie einen Alias (mittels einer »forward-aliasname«-Datei auf master.debian.org verwenden, die in einen halbwegs angenehmen *`ihr-aliasname@debian.org`* umwandelt) oder eine selbstverwaltete Mailingliste auf Alioth benutzen.

Falls Sie entscheiden, dass eine reguläre Mailingliste auf lists.debian.org wirklich das ist, was Sie wollen – nur zu. Füllen Sie eine Anfrage aus und folgen Sie dem HOWTO.

4.2 IRC-Kanäle

Mehrere IRC-Kanäle sind für die Entwicklung von Debian bestimmt. Sie werden hauptsächlich auf dem Open and free technology community (OFTC)-Netzwerk gehostet. Der DNS-Eintrag `irc.debian.org` ist ein Alias für `irc.oftc.net`.

Der Hauptkanal für Debian ist im Allgemeinen `#debian`. Dies ist ein großer Kanal für allgemeine Zwecke, auf dem Benutzer aktuelle Neuigkeiten im Inhalt finden, der von Robotern bereitgestellt wird. `#debian` ist für englischsprachige Nutzer. Es gibt auch `#debian.de`, `#debian-fr`, `#debian-br` und andere Kanäle mit ähnlichen Namen für anderssprachige Nutzer.

Der Hauptkanal für Debian-Entwicklung ist `#debian-devel`. Es ist ein sehr aktiver Kanal; es werden normalerweise mindestens 150 Leute zu jeder Tageszeit dort sein. Es ist ein Kanal für Leute, die an Debian arbeiten, es ist kein Support-Kanal (dafür gibt es `#debian`). Das Themengebiet enthält normalerweise interessante Informationen für Entwickler.

Da `#debian-devel` ein offener Kanal ist, sollten Sie dort nicht über Probleme sprechen, die in debian-private@lists.debian.org diskutiert werden. Es gibt einen anderen Kanal für diesen Zweck. Er heißt `#debian-private` und ist durch einen Schlüssel geschützt. Dieser Schlüssel ist unter `master.debian.org:~debian/misc/irc-password` verfügbar.

Es gibt andere zusätzliche Kanäle die besonderen Themen gewidmet sind. `#debian-bugs` wird für die Koordination von Bug-Squashing-Parties benutzt. `#debian-boot` wird benutzt, um die Arbeit am Debian-Installationsprogramm zu koordinieren. `#debian-doc` wird gelegentlich benutzt, um über die Dokumentation zu reden, wie beispielsweise über das Dokument, das Sie lesen. Andere Kanäle beschäftigen sich mit einer Architektur oder einer Zusammenstellung von Paketen: `#debian-kde`, `#debian-dpkg`, `#debian-jr`, `#debian-edu`, `#debian-oo` (Paket OpenOffice.org) ...

Es existieren außerdem einige nicht englische Entwicklerkanäle, zum Beispiel `#debian-devel-fr` für französchischsprachige Leute, die an der Entwicklung von Debian interessiert sind.

Auch in anderen IRC-Netzwerken existieren Kanäle, die für Debian bestimmt sind, insbesondere auf dem IRC-Netzwerk freenode, auf das bis zum 4.Juni 2006 vom Alias `irc.debian.org` verwiesen wurde.

Um eine Cloak auf Freenode zu bekommen, senden Sie Jörg Jaspert <joerg@debian.org> eine signierte Mail, in der Sie Ihren Nicknamen mitteilen. Schreiben Sie »cloak« irgendwo in die Betreff-Kopfzeilen. Der Nickname sollte registriert werden: Nick Setup Page. Die Mail muss durch einen Schlüssel aus dem Debian-Schlüsselring signiert sein. Lesen Sie bitte die Freenode-Dokumentation, um weitere Informationen über Cloaks zu erhalten.

4.3 Dokumentation

Dieses Dokument enthält viele Informationen, die für Debian-Entwickler nützlich sind, es kann aber nicht alles enthalten. Die meisten anderen interessanten Dokumente sind von der Entwickler-Ecke verlinkt. Nehmen Sie sich

die Zeit all diese Links zu durchstöbern – Sie werden viele weitere Dinge lernen.

4.4 Debian-Maschinen

Debian hat mehrere Computer, die als Server fungieren, meist um kritische Funktionen für das Debian-Projekt bereitzustellen. Die meisten dieser Maschinen werden für Portierungzwecke benutzt und alle haben eine permanente Verbindung ins Internet.

Einige der Maschinen stehen einzelnen Entwicklern so lange zur Verfügung, wie die Entwickler die Regeln befolgen, die in den Debian-Rechner Benutzungsrichtlinien festgelegt wurden.

Allgemein gesprochen, können Sie diese Maschinen nach Belieben für Debian-bezogene Zwecke nutzen. Bitte seien Sie freundlich zu den Systemadministratoren und verbrauchen Sie nicht massenhaft Plattenplatz, Netzwerk-Bandbreite oder CPU ohne zuerst die Zustimmung der Systemadministratoren eingeholt zu haben. Üblicherweise werden diese Maschinen von Freiwilligen betrieben.

Bitte achten Sie darauf, Ihre Debian-Passwörter und SSH-Schlüssel, die auf Debian-Maschinen installiert sind, zu schützen. Vermeiden Sie Methoden zum Anmelden oder Hochladen, die Passwörter unverschlüsselt über das Internet übertragen, wie Telnet, FTP, POP, etc.

Bitte legen Sie kein Material auf Debian-Servern ab, das keinen Bezug zu Debian hat, nicht einmal, wenn Sie eine vorrangige Berechtigung haben.

Die aktuelle liste von Debian-Maschinen ist unter `http://db.debian.org/machines.cgi` verfügbar. Diese Web-Seite enthält Maschinennamen, Kontaktinformationen darüber, wer sich anmelden kann, SSH-Schlüssel etc.

Falls Sie ein Problem mit einer Transaktion auf einem Debian-Server haben und der Ansicht sind, dass die Systemverwalter über dieses Problem informiert werden müssten, können Sie die Liste offener Probleme in der DSA-Warteschlange Ihrer Anfragenverfolgung unter `https://rt.debian.org/` prüfen (Sie können sich als Benutzer »debian« anmelden, sein Passwort ist unter `master.debian.org:~debian/misc/rt-password` verfügbar). Um ein neues Problem zu berichten, senden Sie einfach eine E-Mail an admin@rt.debian.org und stellen Sie sicher, dass der Betreff die Zeichenkette »Debian RT« enthält.

Falls Sie ein Problem mit einem bestimmten Dienst haben, der sich nicht auf die Systemadministration bezieht (wie beispielsweise Pakete, die aus dem Archiv entfernt werden sollen, Vorschläge für die Website, etc.) schreiben Sie einen Fehlerbericht zu einem »Pseudo-Paket«. Informationen darüber, wie Sie Fehlerberichte versenden, finden Sie unter Abschnitt 7.1.

Einige der Hauptserver werden eingeschränkt, aber die Informationen von dort werden auf einen anderen Server gespiegelt.

4.4.1 Der Fehler-Server

`bugs.debian.org` ist der vorschriftsmäßige Ort für die Fehlerdatenbank, das BTS (Bug Tracking System).

Falls Sie statistische Auswertungen oder Verarbeitungen von Debian-Fehlern planen, wäre dies der richtige Ort dafür. Bitte schildern Sie indes Ihre Pläne auf debian-devel@lists.debian.org bevor Sie etwas implementieren, um unnötige Doppelarbeit oder vergeudete Ausführungszeit zu vermeiden.

4.4.2 Der FTP-Master-Server

Auf dem Server `ftp-master.debian.org` liegt die vorschriftsmäßige Kopie des Debian-Archivs. Generell landen auf ftp.upload.debian.org hochgeladene Pakete auf diesem Server, siehe Abschnitt 5.6.

Dieser Server ist eingeschränkt; ein Spiegel ist auf `ries.debian.org` verfügbar.

Probleme mit dem Debian-Archiv müssen generell als Fehler des Pseudo-Pakets `ftp.debian.org` oder per E-Mail an ftpmaster@debian.org gemeldet werden, aber sehen Sie sich auch die Vorgehensweisen in Abschnitt 5.9 an.

4.4.3 Der WWW-Master-Server

Der Haupt-Webserver ist `www-master.debian.org`. Auf ihm liegen die offiziellen Web-Seiten, für die meisten Neulinge »das Gesicht von Debian«.

Falls Sie ein Problem mit dem Debian-Webserver finden, sollten Sie generell einene Fehlerbericht an das Pseudo-Paket `www.debian.org` senden. Denken Sie daran zu prüfen, ob bereits sonst jemand dieses Problem an die Fehlerdatenbank gemeldet hat.

4.4.4 Der Personen-Webserver

`people.debian.org` ist der Server, der für die eigenen Web-Seiten der Entwickler über alles Mögliche mit Bezug zu Debian benutzt wird.

Falls Sie einige Debian-spezifischen Informationen haben, die Sie im Web bereitstellen möchten, können Sie dies realisieren, indem Sie das Material im Verzeichnis `public_html` in Ihrem Home-Verzeichnis auf `people.debian.org` ablegen. Es kann über die URL `http://people.debian.org/~`*`Ihre-Benutzer-ID`*`/` darauf zugegriffen werden.

Sie sollten nur diesen besonderen Ort benutzen, da Datensicherungen davon erstellt werden, was auf anderen Rechnern nicht der Fall wäre.

Der einzige Grund andere Rechner zu benutzen, ist üblicherweise, wenn Sie Materialien veröffentlichen möchten, die den U.S.-Exportbeschränkungen unterliegen. In diesem Fall können Sie Server benutzen, die sich außerhalb der Vereinigten Staaten befinden.

Senden Sie eine E-Mail an debian-devel@lists.debian.org, falls Sie Fragen haben.

4.4.5 Die VCS-Server

Falls Sie ein Versionskontrollsystem für Ihre Arbeit an Debian benutzen möchten, können Sie eines der existierenden auf Alioth gehosteten Depots benutzen oder Sie können ein neues Projekt mit einem VCS-Depot Ihrer Wahl beantragen. Alioth unterstützt CVS (cvs.alioth.debian.org/cvs.debian.org), Subversion (svn.debian.org), Arch (tla/baz, beide auf arch.debian.org), Bazaar (bzr.debian.org), Darcs (darcs.debian.org), Mercurial (hg.debian.org) und Git (git.debian.org). Checken Sie `http://wiki.debian.org/Alioth/PackagingProject` aus, wenn Sie planen, die Pakete in einem VCS-Depot zu verwalten. Weitere Informationen über die von Alioth bereitgestellten Dienste finden Sie unter Abschnitt 4.12.

4.4.6 Chroots auf andere Distributionen

Auf einigen Maschinen sind Chroots auf unterschiedliche Distributionen verfügbar. Sie können sie wie folgt nutzen:

```
vore$ dchroot unstable
Shell in Chroot ausführen: /org/vore.debian.org/chroots/user/unstable
```

In allen Chroots sind die normalen Home-Verzeichnisse der Benutzer verfügbar. Sie können mittels `http://db.debian.org/machines.cgi` herausfinden, welche Chroots verfügbar sind.

4.5 Die Entwicklerdatenbank

Die Entwicklerdatenbank unter `https://db.debian.org/` ist ein LDAP-Verzeichnis zur Verwaltung von Debian-Entwicklereigenschaften. Sie können diese Ressource benutzen, um eine Liste der Debian-Entwickler zu durchsuchen. Ein Teil dieser Informationen ist auch durch den Dienst »finger« auf Debian-Servern verfügbar. Versuchen Sie **finger ihr-benutzername@db.debian.org**, um zu sehen, was er berichtet.

Entwickler können sich in der Datenbank anmelden, um verschiedene Informationen über sich selbst zu ändern, wie beispielsweise:

- die Weiterleitungsadresse für Ihre debian.org-E-Mail
- die Anmeldung zu debian-private
- ob Sie in Urlaub sind
- persönliche Informationen, wie Ihre Adresse, das Land, Längen- und Breitengrad Ihres Wohnortes für die Weltkarte der Entwickler, Telefon- und Faxnummern, IRC-Nickname und Homepage
- Passwort und bevorzugte Shell auf Maschinen des Debian-Projekts

Natürlich kann auf die meisten der Informationen nicht durch die Öffentlichkeit zugegriffen werden. Lesen Sie für weitere Informationen die Dokumentation unter `http://db.debian.org/doc-general.html`.

Entwickler können auch ihre SSH-Schlüssel senden, die für die Authentifizierung auf den offiziellen Debian-Maschinen benutzt werden und sogar neue *.debian.net-DNS-Einträge hinzufügen. Diese Funktionen sind unter `http://db.debian.org/doc-mail.html` dokumentiert.

4.6 Das Debian-Archiv

Die Distribution Debian GNU/Linux besteht aus vielen Paketen (aktuell rund 15000 Quellpakete) und ein paar zusätzlichen Dateien (wie beispielsweise Dokumentation und Images von Installationsmedien).

Hier ist ein Beispielverzeichnisbaum eines kompletten Debian-Archivs:

```
dists/stable/main/
dists/stable/main/binary-amd64/
dists/stable/main/binary-armel/
dists/stable/main/binary-i386/
     ...
dists/stable/main/source/
     ...
dists/stable/main/disks-amd64/
dists/stable/main/disks-armel/
dists/stable/main/disks-i386/
     ...

dists/stable/contrib/
dists/stable/contrib/binary-amd64/
dists/stable/contrib/binary-armel/
dists/stable/contrib/binary-i386/
     ...
dists/stable/contrib/source/

dists/stable/non-free/
dists/stable/non-free/binary-amd64/
dists/stable/non-free/binary-armel/
dists/stable/non-free/binary-i386/
     ...
dists/stable/non-free/source/

dists/testing/
dists/testing/main/
     ...
dists/testing/contrib/
     ...
dists/testing/non-free/
     ...

dists/unstable
dists/unstable/main/
     ...
dists/unstable/contrib/
     ...
dists/unstable/non-free/
     ...

pool/
pool/main/a/
pool/main/a/apt/
     ...
pool/main/b/
pool/main/b/bash/
     ...
pool/main/liba/
pool/main/liba/libalias-perl/
     ...
pool/main/m/
pool/main/m/mailx/
     ...
pool/non-free/f/
pool/non-free/f/firmware-nonfree/
     ...
```

Wie Sie sehen können, enthält das Verzeichnis auf der obersten Ebene die beiden Verzeichnisse `dists/` und `pool/`. Letzteres ist ein »Pool«, in dem sich die Pakete derzeit befinden. Er wird von der Archiv-Verwaltungsdatenbank und den beiliegenden Programmen gehandhabt. Ersteres enthält die Distributionen `stable`, `testing` und `unstable`. Die Dateien `Packages` und `Sources` in den Distributions-Unterverzeichnissen können auf Dateien im Verzeichnis `pool/` verweisen. Der Verzeichnisbaum unterhalb jeder Distribution ist auf die gleiche Art angeordnet. Was im Folgenden für `stable` beschrieben wird, ist gleichermaßen auf die Distributionen `unstable` und `testing` anwendbar.

`dists/stable` enthält drei Verzeichnisse, nämlich `main`, `contrib` und `non-free`.

In jedem Bereich gibt es ein Verzeichnis für Quellpakete (`source`) und ein Verzeichnis für jede unterstützte Architektur (`binary-i386`, `binary-amd64`, etc.).

Der Bereich `main` enthält zusätzliche Verzeichnisse, die die Medien-Images und einige notwendige Teile der Dokumentation, die zum Installieren der Debian-Distribution auf einer speziellen Architektur (`disks-i386`, `disks-amd64`, etc.) benötigt werden.

4.6.1 Abschnitte

Der Abschnitt `main` des Debian-Archivs ist das, was die **offizielle Debian GNU/Linux Distribution** ausmacht. Der Abschnitt `main` ist offiziell, da er alle Richtlinien vollständig erfüllt. Bei den beiden anderen Abschnitten ist dies zu einem unterschiedlichen Grad nicht der Fall . Von daher sind sie **nicht** offizieller Teil von Debian GNU/Linux.

Jedes Paket im Abschnitt »main« muss vollständig die Debian-Richtlinien für Freie Software (DFSG) erfüllen sowie alle anderen Anforderungen des Regelwerks, das im Debian Policy Manual beschrieben wurde. Die DFSG sind Debians Definition von »freier Software«. Prüfen Sie das Debian Policy Manual für Details.

Pakete im Abschnitt `contrib` müssen den DFSG entsprechen, könnten aber an anderen Anforderungen scheitern. Zum Beispiel könnten sie von unfreien Paketen abhängen.

Pakete die nicht die DFSG erfüllen werden in den Abschnitt `non-free` platziert. Diese Pakete werden nicht als Teil der Debian-Distribution betrachtet, obwohl ihre Benutzung ermöglicht und die Infrastuktur (wie die Fehlerdatenbank und die Mailinglisten) für unfreie Pakete bereitgestellt wird.

Das Debian Policy Manual enthält eine genauere Definition dieser drei Abschnitte. Die verhergehende Erläuterung ist nur eine Einführung.

Die Unterteilung in drei Abschnitte auf der obersten Ebene des Archivs ist für all die Leute wichtig, die Debian weitergeben möchten, entweder über FTP-Server im Internet oder auf CD-ROMs: Rechtliche Risiken können vermieden werden, wenn nur die Abschnitte `main` und `contrib` weitergegeben werden. Einige Pakete im Abschnitt `non-free` erlauben zum Beispiel keine kommerzielle Weitergabe.

Andererseits könnte ein CD-ROM-Verkäufer die einzelnen Paketlizenzen der Pakete in `non-free` leicht prüfen und seinen CD-ROMs so viele wie erlaubt hinzufügen. (Da dies von Verkäufer zu Verkäufer stark variiert, können Debian-Entwickler diese Arbeit nicht erledigen.)

Beachten Sie, dass der Begriff Abschnitt auch im Bezug auf Kategorien benutzt wird, um die Organisation und das Durchstöbern verfügbarer Pakete zu vereinfachen, z.B. `admin`, `net`, `utils` etc. Diese Abschnitte (eher Unterabschnitte) existierten irgendwann einmal in der Form von Unterverzeichnissen innerhalb des Debian-Archivs. Heutzutage existieren sie nur noch in den »Section«-Kopfzeilenfeldern der Pakete.

4.6.2 Architekturen

In den Anfangstagen war der Linux-Kernel und somit Debian nur für die Intel-i386-Plattformen (oder höher) verfügbar. Aber als Linux zunehmend populärer wurde, wurde der Kernel auf andere Architekturen portiert und Debian begann diese zu unterstützen. Und als ob die Unterstüzung so vieler Hardware noch nicht genug wäre, entschied Debian, einige Portierungen zu erstellen, die auf anderen Unix-Kerneln beruhten, wie `hurd` und `kfreebsd`.

Debian GNU/Linux 1.3 war nur für `i386` verfügbar. Debian 2.0 wurde für die Architekturen `i386` und `m68k` ausgegeben. Debian 2.1 kam für die Architekturen `i386`, `m68k`, `alpha` und `sparc` daher. Seither ist Debian enorm gewachsen. Debian 6 unterstützt im Ganzen neun Architekturen: `amd64`, `armel`, `i386`, `ia64`, `mips`, `mipsel`, `powerpc`, `s390` und `sparc`) und zwei kFreeBSD-Architekturen (`kfreebsd-i386` und `kfreebsd-amd64`).

Informationen für Entwickler und Anwender über die spezifischen Portierung sind auf der Debian Website Portierungen verfügbar.

4.6.3 Pakete

Es gibt zwei Typen von Debian-Paketen, nämlich `source`- und `binary`-Pakete.

Abhängig vom Format des Quellpakets wird es aus einem oder mehreren Dateien zusätzlich zur zwingend notwendigen `.dsc`-Datei bestehen:

- mit Format »1.0« hat es entweder eine `.tar.gz`-Datei oder sowohl eine `.orig.tar.gz`- als auch eine `.diff.gz`-Datei;
- mit Format »3.0« (quilt), hat es zwingend einen ursrünglichen Tarball mit der Endung `.orig.tar.{gz, bz2, xz}`, optional mehrere `.orig-`*`Komponente`*`.tar.{gz, bz2, xz}`-Debian-Tarbälle;
- mit Format »3.0« (nativ) hat es nur einen einzelnen `.tar.{gz, bz2, xz}`-Tarball.

Falls das Paket speziell für Debian entwickelt wurde und nicht außerhalb von Debian verteilt wird, gibt es dort nur eine `.tar.{gz, bz2, xz}`-Datei, die den Quellcode des Programms enthält, »natives« Quellpaket genannt. Falls ein Paket auch anderswo verteilt wird, wird in der `.orig.tar.{gz, bz2, xz}`-Datei der sogenannte `upstream source code` gespeichert, das ist der Quellcode, vom `upstream maintainer` (oft dem Autor der Software). In diesem Fall enthalten die Dateien `.diff.gz` oder `debian.tar.{gz, bz2, xz}` die Änderungen, die durch den Debian-Betreuer vorgenommen wurden.

Die `.dsc`-Datei listet alle Dateien im Quellpaket auf, zusammen mit den Prüfsummen (**md5sums**) und einigen zusätzlichen Informationen über das Paket (Betreuer, Version etc.).

4.6.4 Distributionen

Das im vorherigen Kapitel beschriebene Verzeichnissystem ist selbst innerhalb der `distribution directories` enthalten. Jede Distribution ist derzeit im Verzeichnis `pool` in der obersten Ebene des Debian-Archivs selbst enthalten.

Zusammenfassend gesagt, hat das Debian-Archiv ein Wurzelverzeichnis auf einem FTP-Server. Auf der Spiegel-Site `ftp.us.debian.org` ist beispielsweise das Debian-Archive selbst in /debian, enthalten, was ein gebräuchlicher Ort dafür ist (ein anderer ist `/pub/debian`).

Eine Distribution umfasst Debian-Quell- und -Binärpakete und die jeweiligen Indexdateien `Sources` sowie `Packages`, die die Kopfzeileninformationen von all diesen Paketen enthalten. Erstere werden im Verzeichnis `pool/` aufbewahrt, während letztere im Verzeichnis `dists/` des Archivs aufbewahrt werden (aus Gründen der Rückwärtskompatibilität).

4.6.4.1 Stable, testing und unstable

Es gibt immer Distributionen mit Namen `stable` (angesiedelt in `dists/stable`), `testing` (angesiedelt in `dists/testing`) und `unstable` (angesiedelt in `dists/unstable`). Dies spiegelt den Entwicklungsprozess des Debian-Projekts wider.

In der Distribution `unstable` wird aktiv entwickelt (daher wird diese Distribution manchmal auch die `development distribution` genannt). Jeder Debian-Entwickler kann jederzeit seine Pakete in dieser Distribution ändern. Daher ändert sich der Inhalt dieser Distribution von Tag zu Tag. Da keine besonderen Anstrengungen unternommen werden, sicherzustellen, dass alles in dieser Distribution funktioniert, ist sie manchmal buchstäblich instabil.

Die Distribution testing wird automatisch aus Paketen von `unstable` erzeugt, falls sie bestimmte Voraussetzungen erfüllen. Diese Voraussetzungen sollten eine gute Qualität für Pakete innerhalb von `testing` gewährleisten. Die Aktualisierung von `testing` wird zweimal täglich lanciert, gleich nachdem die neuen Pakete installiert wurden. Siehe Abschnitt 5.13.

Nach einer Entwicklungsperiode, sobald der Veröffentlichungsverwalter sie als tauglich erachtet, wird die Distribution `testing` eingefroren. Das bedeutet, dass die Richtlinien, die steuern welche Pakete von `unstable` nach `testing` verschoben werden, verschärft werden. Pakete, die zu viele Fehler aufweisen, werden entfernt. In `testing` sind außer Fehlerkorrekturen keine Änderungen erlaubt. Nachdem, abhängig vom Fortschritt, einige Zeit verstrichen ist, wird die Distribution `testing` sogar noch weiter eingefroren. Einzelheiten darüber, wie die Distribution `testing` gehandhabt wird, werden vom Veröffentlichungs-Team auf »debian-devel-announce« publiziert. Nachdem die offenen Probleme zur Zufriedenheit des Veröffentlichungs-Teams gelöst wurden, wird die Distribtion veröffentlicht. Veröffentlichen bedeutet, dass `testing` in `stable` umbenannt wird und für das neue `testing` eine neue Kopie erstellt wird. Das vorherige `stable` wird in `oldstable` umbenannt und bleibt bis zur endgültigen Archivierung dort. Bei der Archivierung wird der Inhalt nach `archive.debian.org` verschoben.

Der Entwicklungszyklus hängt von der Einschätzung ab, ob die Distribution `unstable` nach einer Periode, in der sie `testing` durchläuft, `stable` geworden ist.Sogar wenn eine Distribution stabil geworden ist, verbleiben zwangsläufig ein paar Fehler – daher wird die stabile Distribution ab und zu aktualisiert. Diese Aktualisierungen wurden jedoch sehr gründlich getestet und werden in einem einzelnen Archiv eingeführt, um das Risiko

zu vermindern, neue Fehler einzuschleppen. Sie können die geplanten Ergänzungen zu stable im Verzeichnis proposed-updates finden. Diese Pakete in proposed-updates, die den Anforderungen entsprechen, werden periodisch in einem Stapellauf in die Stable-Distribution verschoben und die Überarbeitungsstufe der Stable-Distribution wird erhöht (z.B. »6.0« wird »6.0.1«, »5.0.7« wird »5.0.8« und so weiter). Bitte sehen Sie unter uploads to the stable distribution nach, um weitere Einzelheiten zu erfahren.

Beachten Sie, dass die Entwicklung unter unstable während der Periode des Einfrierens weitergeht, da die Distribution unstable an der Stelle parallel mit testing verbleibt.

4.6.4.2 Weitere Informationen über die Testing-Distribution

Pakete werden üblicherweise in der Distribution testing installiert, nachdem sie in gewissem Maße in unstable Tests unterzogen wurden.

Lesen Sie bitte die Informationen über die Testing-Distribution, um weitere Einzelheiten zu erfahren.

4.6.4.3 Experimental

Die Distribution experimental ist eine Spezialdistribution. Sie ist keine vollständige Distribution im Sinn von stable, testing und unstable. Stattdessen ist sie als temporärer Sammelpunkt für hoch experimentelle Software gedacht, bei der eine gute Chance besteht, das ganze System zu zerstören oder von Software, die nur zu instabil ist, sogar für die Distribution unstable (bei der es aber dennoch einen Grund gibt, sie zu paketieren). Von Benutzern, die Pakete aus experimental herunterladen und installieren, wird erwartet, dass sie ausreichend gewarnt wurden. Kurz gesagt, ist in der Distribution experimental alles möglich.

Dies sind die Zeilen der sources.list(5) für experimental:

```
deb http://ftp.xy.debian.org/debian/ experimental main
deb-src http://ftp.xy.debian.org/debian/ experimental main
```

Falls eine Chance besteht, dass die Software ein System schwer beschädigen könnte, ist es wahrscheinlich besser, sie in experimental abzulegen. Zum Beispiel sollte ein experimentell komprimiertes Dateisystem wahrscheinlich nach experimental wandern.

Jedes Mal, wenn eine neue Originalversion eines Pakets vorliegt, das neue Funktionen mitbringt, aber ältere beschädigt, sollte es entweder nicht oder nach experimental hochgeladen werden. Eine neue Beta-Version einer Software, die eine völlig unterschiedliche Konfiguration nutzt, kann nach Ermessen des Betreuers nach experimental wandern. Falls Sie an einer inkompatiblen oder komplexen Situation bei Upgrades arbeiten, können sie experimental auch als Sammelpunkt benutzen, damit Tester frühzeitig Zugriff darauf haben.

Einige experimentelle Software kann immer noch, mit ein paar Warnungen in der Beschreibung, nach unstable wandern, dies wird aber nicht empfohlen, da von Paketen aus unstable erwartet wird, dass sie sich nach testing und dann nach stable ausbreiten. Sie sollten keine Angst haben experimental zu benutzen, da es dem FTP-Master keine Mühe macht. Die experimentellen Pakete werden reglemäßig entfernt, wenn Sie das Paket mit einer höheren Versionsnummer nach unstable hochladen.

Neue Software, die das System voraussichtlich nicht schädigt, kann direkt in unstable wandern.

Eine Alternative zu experimental ist die Benutzung Ihres persönlichen Webspaces auf people.debian.org.

4.6.5 Codenamen der Veröffentlichungen

Jede veröffentlichte Debian-Distribution hat einen code name: Debian 1.1 wird buzz genannt, Debian 1.2 rex, Debian 1.3 bo, Debian 2.0 hamm, Debian 2.1 slink, Debian 2.2 potato, Debian 3.0 woody, Debian 3.1 sarge, Debian 4.0 etch, Debian 5.0 lenny, Debian 6.0, squeeze und die nächste Veröffentlichung wird wheezy genannt. Es gibt außerdem eine »Pseudo-Distribution« mit dem Namen sid, die der aktuellen Distribution unstable entspricht. Da Pakete von unstable nach testing wandern, wenn sie stabil werden, wird sid selbst nie veröffentlicht. Ebenso wie die üblichen Inhalte der Debian-Distribution, enthält sid Pakete für Architekturen, die noch nicht offiziell durch Debian unterstützt werden. Es ist geplant, diese Architekturen an irgendeinem zukünftigen Datum in die Hauptdistribution zu integrieren.

Da Debian ein offenes Entwicklungsmodell hat (d.h. jeder kann teilnehmen und der Entwicklung folgen), werden sogar die Distributionen unstable und testing über das Internet durch die Debian-FTP- und -HTTP-Netzwerkserver verteilt. Folglich müsste, falls das Verzeichnis, das die Kandidatenversion für die Veröffentlichung enthält, testing genannt würde, es beim Veröffentlichen der Version in stable umbenannt werden, was dazu führen würde, dass alle FTP-Spiegel die ganze Distribution erneut erhalten müssten (die ziemlich groß ist).

Wären andererseits von Anfang an die Distributionsverzeichnisse `Debian-x.y` genannt worden, würden die Leute denken, die Debian-Veröffentlichung `x.y` sei verfügbar. (Dies geschah früher, als ein CD-Verkäufer eine Debian 1.0 CD-ROM erstellte, die auf einer pre-1.0-Entwicklerversion basierte. Das ist der Grund, weshalb die erste offizielle Debian-Veröffentlichung 1.1 und nicht 1.0 war.)

Dadurch werden die Namen der Distributionsverzeichnisse im Archiv durch ihre Codenamen festgelegt und nicht durch ihren Veröffentlichungsstatus. (z.B. »squeeze«). Diese Namen bleiben während der Entwicklungszeit und nach der Veröffentlichung erhalten. Symbolische Links, die einfach geändert werden können, geben die aktuell veröffentlichte stabile Distribution an. Das ist der Grund, weshalb die echten Distributionsverzeichnisse `code names` benutzen, während symbolische Links für `stable`, `testing` und `unstable` auf die entsprechenden Veröffentlichungsverzeichnisse verweisen.

4.7 Debian-Spiegel

Für die verschiedenen Archive zum Herunterladen und die Website stehen mehrere Spiegel zur Verfügung, um die vorschriftsmäßigen Server vor großer Auslastung zu bewahren. Eigentlich sind einige der vorschriftsmäßigen Server nicht öffentlich zugänglich – eine erste Schicht von Spiegeln balanciert stattdessen die Last aus. Auf diese Art greifen die Anwender immer auf die Spiegel zu und sind es gewöhnt sie zu benutzen. Dies ermöglich es Debian, seine Bandbreitenanforderungen besser über mehrere Server und Netzwerke zu verteilen und vermeidet grundsätzlich, dass Benutzer auf den einen primären Ort aufschlagen. Beachten Sie, dass die erste Schicht von Spiegelservern so aktuell wie möglich ist, da ihre Aktualisierung durch die internen Sites ausgelöst wird (dies wird »push mirroring« genannt).

All die Informationen über Debian-Spiegel, einschließlich der verfügbaren FTP-/HTTP-Server, können unter `http://www.debian.org/mirror/` gefunden werden. Diese nützliche Seite enthält auch Informationen und Werkzeuge, die hilfreich sein können, falls Sie daran interessiert sind, entweder für internen oder öffentlichen Zugriff, Ihren eigenen Spiegel einzurichten.

Beachten Sie, dass Spiegel generell durch Dritte betrieben werden, die ein Interesse daran haben, Debian zu helfen. Daher haben Entwickler generell keine Konten auf diesen Maschinen.

4.8 Das Eingangssystem

Das Eingangssystem ist dafür verantwortlich aktualisierte Pakete zu sammeln und im Debian-Archiv zu installieren. Es besteht aus einer Zusammenstellung von Verzeichnissen und Skripten, die auf `ftp-master.debian.org` installiert sind.

Pakete werden durch alle Betreuer in ein Verzeichnis hochgeladen, das `UploadQueue` heißt. Dieses Verzeichnis wird alle paar Minuten durch einen Daemon gescannt, der **queued** genannt wird, es werden `*.command`-Dateien ausgeführt und verbleibende, korrekt signierte `*.changes`-Dateien werden zusammen mit ihren zugehörigen Dateien in das Verzeichnis `unchecked` verschoben. Dieses Verzeichnis ist für die meisten Entwickler unsichtbar, da FTP-Master eingeschränkt wurde. Es wird alle 15 Minuten durch das Skript **dak process-upload** gelesen, das die Richtigkeit der hochgeladenen Pakete und ihre kryptografischen Signaturen prüft. Falls das Paket für installationsbereit befunden wird, wird es in das Verzeichnis `done` verschoben. Falls dies das erste Hochladen des Pakets ist (oder es neue Binärpakete enthält), wird es in das Verzeichnis `new` verschoben, wo es auf die Freigabe durch die FTP-Master wartet. Falls das Paket Dateien enthält, die manuell installiert werden müssen, wird es in das Verzeichnis `byhand` verschoben, wo es auf die manuelle Installation durch die FTP-Master wartet. Andernfalls, falls ein Fehler aufgespürt wurde, wird das Paket abgewiesen und landet im Verzeichnis `reject`.

Sobald das Paket akzeptiert wurde, sendet das System dem Betreuer per E-Mail eine Bestätigung, schließt alle Fehler, die durch das Hochladen als behoben markiert wurden und die Auto-Builder können beginnen, es erneut zu kompilieren. Das Paket ist nun öffentlich unter `http://incoming.debian.org/` zugänglich, bis es wirklich im Debian-Archiv installiert wird. Dies geschieht viermal täglich (und wird aus historischen Gründen »dinstall run« genannt). Das Paket wird dann aus dem Eingangssystem entfernt und zusammen mit den anderen Paketen in den Pool installiert. Sobald alle anderen Aktualisierungen (Erzeugen neuer `Packages`- und `Sources`-Indexdateien zum Beispiel) durchgeführt wurden, wird ein Spezialskript aufgerufen, das alle Primärspiegel auffordert, sich selbst zu aktualisieren.

Die Archivverwaltungs-Software wird außerdem die OpenPGP-/GnuPG-signierte `.changes`-Datei senden, die Sie an die entsprechenden Mailinglisten gesandt haben. Falls ein Paket veröffentlicht wird, bei dem die Distribution auf `stable` gesetzt ist, wird die Ankündigung an debian-changes@lists.debian.org gesandt. Falls ein Paket veröffentlicht wird, bei dem die Distribution auf `unstable` oder `experimental` gesetzt ist, wird die Ankündigung stattdessen an debian-devel-changes@lists.debian.org gesandt.

Obwohl der FTP-Master eingeschränkt wurde, ist eine Kopie für alle Entwickler unter `ries.debian.org` verfügbar.

4.9 Paketinformationen

4.9.1 Im Web

Jedes Paket hat mehrere zugehörige Web-Seiten. `http://packages.debian.org/`*`Paketname`* zeigt jede Version des Pakets, die in verschiedenen Distributionen verfügbar ist. Jede Version verweist auf eine Seite, die Informationen einschließlich der Paketbeschreibung, der Abhängigkeiten und der Links zum Herunterladen, bereitstellt.

Die Fehlerdatenbank verfolgt Fehler für jedes Paket. Sie können die Fehler unter der URL `http://bugs.debian.org/`*`Paketname`* ansehen.

4.9.2 Das Hilfswerkzeug dak ls

dak ls ist Teil der Werkzeugsammlung Dak, die verfügbare Paketversionen für alle bekannten Distributionen und Architekturen auflistet. Das Werkzeug **dak** ist auf `ftp-master.debian.org` und auf dem Spiegel `ries.debian.org` verfügbar. Es benutzt ein einziges Argument, das einem Paketnamen entspricht. Ein Beispiel erklärt es besser:

```
$ dak ls evince
evince | 0.1.5-2sarge1 |     oldstable | source, alpha, arm, hppa, i386, ia64,  ↩
    m68k, mips, mipsel, powerpc, s390, sparc
evince |    0.4.0-5 |     etch-m68k | source, m68k
evince |    0.4.0-5 |        stable | source, alpha, amd64, arm, hppa, i386, ia64 ↩
    , mips, mipsel, powerpc, s390, sparc
evince |   2.20.2-1 |       testing | source
evince | 2.20.2-1+b1 |       testing | alpha, amd64, arm, armel, hppa, i386, ia64 ↩
    , mips, mipsel, powerpc, s390, sparc
evince |   2.22.2-1 |      unstable | source, alpha, amd64, arm, armel, hppa,  ↩
    i386, ia64, m68k, mips, mipsel, powerpc, s390, sparc
```

An diesem Beispiel können Sie sehen, dass sich die Version in `unstable` von der Version in `testing` unterscheidet und dass dort ein rein binärer NMU des Pakets für alle Architekturen durchgeführt wurde. Jede Version des Pakets wurde auf allen Architekturen neu kompiliert.

4.10 Das Paketverfolgunssystem

Das Paketverfolgungssystem (Package Tracking System/PTS) ist ein E-Mail-basiertes Werkzeug, das die Aktivität eines Quellpakets verfolgt. Dies bedeutet tatsächlich, dass Sie die gleichen E-Mails wie der Paketbetreuer erhalten, indem Sie einfach sich einfach im PTS beim Paket einschreiben.

Jede E-Mail, die durch das PTS versandt wird, ist unter einem der nachfolgenden Schlüsselwörter aufgelistet. Dies wird Ihnen sie Auswahl erleichtern, welche E-Mails Sie erhalten möchten.

Standardmäßig erhalten Sie:

`bts` alle Fehlerberichte und folgenden Diskussioen

`bts-control` die E-Mail-Benachrichtigungen von control@bugs.debian.org über Statusänderungen von Fehlermeldungen

`upload-source` die E-Mail-Benachrichtigung von **dak**, wenn ein hochgeladenes Quellpaket akzeptiert wird

`katie-other` andere Warn- und Fehler-E-Mails von **dak** (wie beispielsweise Unterschiedlichkeit beim Überschreiben eines Abschnitts oder eines Prioritätsfelds).

`buildd` Benachrichtigungen über das Fehlschlagen der Paketerstellung, gesandt durch das Netzwerk der Paketerstellungs-Daemons. Sie enthalten für Analysen einen Zeiger auf die Erstellungsprotokolle.

default jede nicht automatische E-Mail, die durch Leute an das PTS geschickt wird, die Kontakt zu den Abonnenten des Pakets aufnehmen möchten. Dies kann durch Senden einer E-Mail an *Quellpaket*@packages.qa.debian.org erledigt werden. Um Spam zu vermeiden, müssen alle Nachrichten, die an diese Adressen gesandt werden, die Kopfzeile X-PTS-Approved mit einem nicht leeren Wert enthalten.

contact Mails, die dem Betreuer mittels der *@packages.debian.org-E-Mail-Aliase gesandt werden.

summary regelmäßige Zusammenfassungs-E-Mails über den Status des Pakets, einschließlich Fortschritt in testing, DEHS-Benachrichtigungen von neuen Originalversionen und einer Benachrichtigung, falls das Paket entfernt oder verwaist wird.

Sie können sich außerdem entscheiden, zusätzliche Informationen zu erhalten:

upload-binary die E-Mail-Benachrichtigung von **katie**, wenn ein hochgeladenes Binärpaket akzeptiert wurde. Mit anderen Worten – Sie können immer, wenn ein Paketerstellungs-Daemon oder jemand, der Ihr Paket portiert und für eine andere Architektur hochlädt, eine E-Mail erhalten, um zu verfolgen, wie Ihr Paket für alle Architekturen neu kompiliert wird.

cvs »commit«-Benachrichtigungen des VCS, falls das Paket ein VCS-Depot hat und der Betreuer im PTS eine Weiterleitung von »commit«-Benachrichtigungen eingerichtet hat. Der Name »cvs« ist historisch bedingt, in den meisten Fällen werden »commit«-Benachrichtigungen von anderen VCS, wie Subversion oder Git kommen.

ddtp Übersetzungen und Beschreibungen von Debconf-Schablonen, die an das »Debian Description Translation Project« gesandt wurden.

derivatives Informationen über Änderungen am Paket, die in abgeleiteten Distributionen (zum Beispiel Ubuntu) vorgenommen wurden.

derivatives-bugs Fehlerberichte und Kommentare von abgeleiteten Distributionen (zum Beispiel Ubuntu)

4.10.1 Die E-Mail-Schnittstelle des PTS

Sie können Ihr(e) Abonnement(s) zum PTS steuern, indem Sie verschiedene Befehle an pts@qa.debian.org senden.

subscribe <sourcepackage> [<email>] abonniert *E-Mails* zu Kommunikationen, die sich auf das Quellpaket *Quellpaket* beziehen. Falls das zweite Argument fehlt, wird die Absenderadresse benutzt. Wenn *Quellpaket* kein gültiges Quellpaket ist, werden Sie eine Warnung erhalten. Falls es jedoch um ein gültiges Binärpaket handelt, wird das PTS Ihnen das entsprechende Quellpaket abonnieren.

unsubscribe <sourcepackage> [<email>] entfernt ein vorheriges Abonnement des Quellpakets *Quellpaket* unter Benutzung der angegebenen E-Mail-Adresse oder der Absenderadresse, falls das zweite Argument ausgelassen wird.

unsubscribeall [<email>] entfernt alle Abonnements der angegebenen E-Mail-Adresse oder der Absenderadresse, falls das zweite Argument ausgelassen wurde.

which [<email>] listet alle Abonnements des Absenders oder der optional angegebenen E-Mail-Adresse auf.

keyword [<email>] zählt die Schlüsselwörter auf, die Sie akzeptieren. Eine Erklärung der Schlüsselwörter finden Sieoberhalb. Hier ist eine kurze Zusammenfassung:

- bts: E-Mails von der Debian-Fehlerdatenbank
- bts-control: Antworten auf E-Mails, die an control@bugs.debian.org gesandt wurden
- summary: automatische Zusammenfassungs-E-Mails über den Status eines Pakets
- contact: E-Mails, die dem Betreuer durch die *@packages.debian.org-Aliase gesandt werden
- cvs: Benachrichtigungen über VCS-Commits
- ddtp: Übersetzungen von Beschreibungen und Debconf-Schablonen
- derivatives: Änderungen am Paket durch abgeleitete Distributionen
- derivatives-bugs: Fehlerberichte und Kommentare von abgeleiteten Distributionen
- upload-source: Ankündigung, dass eine neu hochgeladene Quelle akzeptiert wurde

- `upload-binary`: Ankündigung eines neuen, rein binären Hochladens (Poertierung)
- `katie-other`: andere E-Mails von FTP-Mastern (Ungleichheit beim Überschreiben etc.)
- `buildd`: Benachrichtigungen über das Fehlschlagen der Paketerstellung, gesandt durch Paketerstellungs-Daemons
- `default`: all die anderen E-Mails (die nicht automatisch erstellt wurden)

`keyword <sourcepackage> [<email>]` entspricht dem vorherigen Element, aber für das angegebene Quellpaket, da Sie eine andere Zusammenstellung von Schlüsselwörtern für jedes Quellpaket wählen könnten.

`keyword [<email>] {+|-|=} <list of keywords>` E-Mails akzeptieren (+) oder ablehnen (-), die unter dem/den angegebenen Schlüsselwort/Schlüsselwörtern klassifiziert wurden. Definiert die Liste (=) akzeptierter Schlüsselwörter. Dies ändert die Standardzusammenstellung der durch einen Anwender akzeptierten Schlüsselwörter.

`keywordall [<email>] {+|-|=} <list of keywords>` E-Mails akzeptieren (+) oder ablehnen (-), die unter dem/den angegebenen Schlüsselwort/Schlüsselwörtern klassifiziert wurden. Definiert die Liste (=) akzeptierter Schlüsselwörter. Dies ändert die Zusammenstellung der durch einen Anwender akzeptierten Schlüsselwörter aller derzeit aktiven Abonnements.

`keyword <sourcepackage> [<email>] {+|-|=} <list of keywords>` entspricht dem vorherigen Element, überschreibt aber die Liste der Schlüsselwörter für das angegebene Quellpaket

`quit | thanks | --` stoppt die Verarbeitung von Befehlen. Alle folgenden Zeilen werden vom Roboter ignoriert.

Das Befehlszeilen-Hilfswerkzeug **pts-subscribe** (aus dem Paket `devscripts`) kann praktisch sein, um temporär einige Paket zu abonnieren, zum Beispiel nach einem NMU (Hochladen durch jemanden, der nicht Betreuer des Pakets ist).

4.10.2 PTS-E-Mails filtern

Sobald Sie ein Paket abonniert haben, werden Sie die E-Mails bekommen, die an das *`Quellpaket`*`@packages.qa.debian.org` geschickt werden. An diese E-Mails sind besondere Kopfzeilen angehängt, die Ihnen ermöglichen, sie per Filter in ein besonderes Postfach abzulegen (z.B. mit **procmail**). Die hinzugefügten Kopfzeilen sind `X-Loop`, `X-PTS-Package`, `X-PTS-Keyword` und `X-Unsubscribe`.

Hier folgt ein Beispiel hinzugefügter Kopfzeilen für die Benachrichtigung, dass die Quelle des Pakets `dpkg` hochgeladen wurde:

```
X-Loop: dpkg@packages.qa.debian.org
X-PTS-Package: dpkg
X-PTS-Keyword: upload-source
List-Unsubscribe: <mailto:pts@qa.debian.org?body=unsubscribe+dpkg>
```

4.10.3 VCS-Commits in das PTS weiterleiten

Falls Sie zur Verwaltung eines Debian-Pakets ein VCS-Depot mit öffentlichem Zugriff verwenden, möchten Sie vielleicht die Commit-Benachrichtigungen an das PTS weiterleiten, damit die Abonnenten (und möglicherweise Mitbetreuer) die Entwicklung des Pakets genau verfolgen können.

Sobald Sie das VCS-Depot eingerichtet haben, um Commit-Benachrichtigungen zu erzeugen, müssen Sie nur sicherstellen, dass es eine Kopie dieser E-Mails an *`Quellpaket`*`_cvs@packages.qa.debian.org` sendet. Nur die Leute, die das Schlüsselwort `cvs` akzeptieren werden diese Benachrichtigungen erhalten. Beachten Sie, dass die E-Mail von einer `debian.org`-Maschine versandt werden muss. Andernfalls müssen Sie die Kopfzeile `X-PTS-Approved:1` hinzufügen.

Für Subversion-Depots wird die Benutzung von Svnmailer empfohlen. Ein Beispiel, wie das funktioniert, finden Sie unter http://wiki.debian.org/Alioth/PackagingProject

4.10.4 Die PTS-Web-Schnittstelle

Das PTS hat eine Web-Schnittstelle unter http://packages.qa.debian.org/, die eine große Menge Informationen über jedes Quellpaket zusammenträgt. Sie zeichnet sich durch viele nützliche Links aus (BTS, QA-Statistiken, Kontaktinformationen, DDTP-Übersetzungsstatus, Paketerstellungsprotokolle) und sammelt viel mehr Informationen von verschiedenen Stellen (30 letzte Einträge des Änderungsprotokolls, Teststatus etc.) Es ist ein sehr nützliches Werkzeug, falls Sie wissen möchten, was bei einem speziellen Quellpaket vorgeht. Außerdem gibt es dort ein Formular, das Ihnen erlaubt einfach Abonnements des PTS mittels E-Mail einzurichten.

Sie können mit einer URL wie `http://packages.qa.debian.org/`*`Quellpaket`* direkt zur Web-Seite springen, die eine spezielles Quellpaket betrifft.

Diese Web-Schnittstelle wurde gestaltet wie ein Portal für die Entwicklung von Paketen: Sie können Ihren Paketseiten benutzerdefinierten Inhalt hinzufügen (Nachrichtenelemente, die auf unbestimmte Zeit verfügbar bleiben sollen) sowie Nachrichtenelemente im letzten Nachrichtenabschnitt.

Statische Nachrichtenelemente können folgendes anzeigen:

- die Verfügbarkeit eines auf Alioth gehosteten Projekts für die Mitbetreuung des Pakets
- einen Link zu der Ursprungs-Website
- einen Link zur Fehlerverfolgung der Originalautoren
- die Existenz eines IRC-Kanals, der sich dieser Software widmet
- jede andere verfügbare Ressource, die nützlich zur Verwaltung des Pakets sein könnte

Üblicherweise können Nachrichtenelemente benutzt werden, um Folgendes anzukündigen:

- Beta-Pakete sind zum Testen verfügbar.
- Fertige Pakete werden nächsten Woche erwartet.
- Das Paketieren wird gerade von Grund auf neu gemacht.
- Es sind Backports verfügbar
- Der Paketbetreuer ist in Urlaub (falls es gewünscht ist, diese Information zu veröffentlichen).
- Es wird an einem NMU gearbeitet.
- Etwas wichtiges wird das Paket beeinflussen.

Beide Arten von Nachrichten werden auf ähnliche Weise erzeugt: Sie müssen lediglich eine E-Mail an pts-static-news@qa.debian.org oder an pts-news@qa.debian.org senden. Die E-Mail sollte angeben, welches Paket sie betrifft, indem der Name des Quellpakets in einer `X-PTS-Package`-E-Mail-Kopfzeile oder in einer `Package`-Pseudo-Kopfzeile (wie in einem Fehlerbericht) angegeben wird. Falls eine URL in der E-Mail-Kopfzeile `X-PTS-Url` oder der Pseudo-Kopfzeile `Url` verfügbar ist, dann ist das Ergebnis ein Link auf diese URL, anstatt eines vollständigen Nachrichtenelements.

Hier folgen ein paar Beispiele gültiger E-Mails, die benutzt werden, um Nachrichtenelemente im PTS zu generieren. Das Erste fügt der Viewsvn-Schnittstelle der Debian-CD im Abschnitt »statische Information« einen Link hinzu:

```
From: Raphael Hertzog <hertzog@debian.org>
To: pts-static-news@qa.debian.org
Subject: Browse debian-cd SVN repository

Package: debian-cd
Url: http://svn.debian.org/viewsvn/debian-cd/trunk/
```

Das Zweite ist eine Ankündigung, die an eine Mailingliste gesandt wird, die außerdem an das PTS geschickt wird, so dass sie auf der PTS-Web-Seite des Pakets veröffentlicht wird. Beachten Sie die Benutzung des BCC-Feldes, um zu vermeiden, dass Antworten fälschlicherweise an das PTS gesandt werden.

```
From: Raphael Hertzog <hertzog@debian.org>
To: debian-gtk-gnome@lists.debian.org
Bcc: pts-news@qa.debian.org
Subject: Galeon 2.0 backported for woody
X-PTS-Package: galeon

Hello gnomers!

I'm glad to announce that galeon has been backported for woody. You'll find
everything here:
...
```

Überlegen Sie es sich gut, bevor Sie dem PTS ein Nachrichtenelement hinzufügen, da Sie es später weder löschen noch bearbeiten können. Das einzige, was Sie tun können, ist, ein zweites Nachrichtenelement zu senden, das die Information des vorherigen missbilligt.

4.11 Paketübersicht des Entwicklers

Ein Webportal der QS (Qualitätssicherung) ist unter http://qa.debian.org/developer.php verfügbar. Es zeigt eine tabellarische Auflistung aller Pakete eines einzelnen Entwicklers an (einschließlich derer, bei denen eine Gemeinschaft als Mitbetreuer aufgelistet ist). Die Tabelle gibt einen guten Überblick über die Pakete des Betreuers: Anzahl der Fehler nach Schweregrad, Liste der verfügbaren Versionen in jeder Distribution, Test-Status und vieles mehr, einschließlich Links zu irgendwelchen anderen nützlichen Informationen.

Es ist ein guter Tipp, regelmäßig auf seine eigenen Daten zu schauen, um keine offenen Fehler oder Zuständigkeiten für Pakete zu vergessen.

4.12 Debians FusionForge-Installation: Alioth

Alioth ist ein Debian-Dienst, der auf einer geringfügig veränderten Version der FusionForge-Software beruht (die sich aus SourceForge und GForge entwickelt hat). Diese Software bietet Entwicklern Zugriff auf einfach benutzbare Werkzeuge, wie Paketverfolgung, Patch-Verwaltung, Projekt-/Aufgabenverwaltung, Speicherplatzbereitstellung, Mailinglisten, VCS-Depots etc. All diese Werkzeuge werden über eine Web-Schnittstelle verwaltet.

Es ist dazu gedacht, Hilfmittel für freie Softwareprojekte bereitzustellen, die von Debian unterstützt oder geleitet werden, Beiträge externer Entwickler zu Projekten, die von Debian begonnen wurden zu fördern und bei Projekten zu helfen, deren Ziele von Debian oder dessen Ableitungen gefördert werden. Es wird durch viele Debian-Teams stark frequentiert und stellt Speicherplatz für alle Arten von VCS-Depots bereits.

Alle Debian-Entwickler haben automatisch ein Konto auf Alioth. Sie können es über die Passwort-Wiederherstellungseinrichtung aktivieren. Externe Entwickler können um Gastkonten bitten.

Um weitere Informationen zu erhalten, besuchen Sie die folgenden Links:

- http://wiki.debian.org/Alioth
- http://wiki.debian.org/Alioth/FAQ
- http://wiki.debian.org/Alioth/PackagingProject
- http://alioth.debian.org/

4.13 Annehmlichkeiten für Entwickler

4.13.1 LWN-Abonnements

Seit Oktober 2002 hat HP für alle interessierten Debian-Entwickler ein LWN-Abonnement gesponsort. Einzelheiten, wie man Zugriff auf diesen Vorteil erhält, finden Sie unter http://lists.debian.org/debian-devel-announce/2002/10/msg00018.html.

4.13.2 Gandi.net-Rabatt für Speicherplatzbereitstellung

Seit November 2008 bietet Gandi.net Debian-Entwicklern einen Rabatt für seine VPS-Speicherplatzbereitstellung. Siehe http://lists.debian.org/debian-devel-announce/2008/11/msg00004.html.

Kapitel 5

Pakete verwalten

Dieses Kapitel enthält Informationen, die sich auf das Erstellen, Hochladen Verwalten und Portieren von Paketen beziehen.

5.1 Neue Pakete

Falls Sie eine neues Paket für die Debian-Distribution erstellen möchten, sollten Sie zuerst die Liste der Arbeitbedürfenden und voraussichtlichen Pakete (WNPP) prüfen. Die Prüfung der WNPP-Liste stellt sicher, dass nicht bereits jemand an der Paketierung dieser Software arbeitet und kein doppelter Aufwand betrieben wird. Weitere Informationen finden Sie auf den WNPP-Web-Seiten.

Ausgehend von der Annahme, dass noch niemand an Ihrem zukünftigen Paket arbeitet, müssen Sie einen Fehlerbericht (Abschnitt 7.1) zu dem Pseudopaket `wnpp` senden, in dem Sie Ihren Plan vorstellen, ein neues Paket zu erzeugen, einschließlich, aber nicht nur auf eine Beschreibung des Pakets beschränkt, der Lizenz des zukünftigen Pakets und der derzeitigen URL, von der es heruntergeladen werden kann.

Sie sollten den Betreff des Fehlers auf `ITP: `*`Paketname --kurze Beschreibung`* setzen, wobei Sie *`Paketname`* durch den Namen Ihres Pakets ersetzen. Der Schweregrad des Fehlerberichts sollte auf `wishlist` gesetzt werden. Bitte senden Sie mit der Kopfzeile X-Debbugs-CC eine Kopie an debian-devel@lists.debian.org (benutzen Sie nicht CC:, da in diesem Fall der Betreff der Nachricht die Fehlernummer nicht angibt). Falls Sie so viele (mehr als zehn) neue Pakete paketieren, dass die Benachrichtigung auf der Liste als störend empfunden würde, senden Sie stattdessen nach dem Einreichen des Fehlers eine Zusammenfassung an die Liste »debian-devel«. Dies wird andere Entwickler über die bevorstehenden Pakete informieren und eine Überprüfung Ihrer Beschreibung und Ihres Paketnamens ermöglichen.

Bitte fügen Sie im Änderungsprotokoll des neuen Pakets den Eintrag `Closes: #`*`nnnnn`* hinzu, um den Fehlerbericht automatisch zu schließen, sobald das neue Paket im Archiv installiert wird (siehe Abschnitt 5.8.4).

Falls Sie der Ansicht sind, Ihr Paket bedürfe einiger Erklärungen für die Administratoren der Paketwarteschlange NEW, so fügen Sie diese dem Änderungsprotokoll hinzu, senden Sie die E-Mail, die Sie als Betreuer nach dem Upload zurückbekommen haben an ftpmaster@debian.org oder antworten Sie auf die ablehnende E-Mail, im Fall dass Sie bereits erneut hochladen.

Wenn sicherheitskritische Fehler geschlossen werden, fügen Sie die CVE-Nummern sowie `Closes: #`*`nnnnn`* bei. Dies ist nützlich zur Verfolgung von Schwachstellen durch das Sicherheits-Team. Falls etwas hochgeladen wird, bevor die Warnungs-ID bekannt ist, sollte beim nächsten Upload der historische Änderungsprotokolleintrag geändert werden. Bitte fügen Sie sogar in diesem Fall dem Original-Änderungsprotokolleintrag alle verfügbaren Hinweise auf Hintergrundinformationen hinzu.

Es gibt eine Vielzahl von Gründen, weshalb Paketbetreuer um die Ankündigung ihrer Absichten gebeten werden:

- Es hilft dem (möglicherweise neuen) Betreuer, die Erfahrung der Leute auf der Liste anzuzapfen und teilt ihm mit, ob jemand anderes bereits daran arbeitet.
- Es zeigt anderen Leuten, die darüber nachdenken am Paket zu arbeiten, dass es bereits einen Freiwilligen gibt, so dass der Aufwand verteilt werden kann.
- Es sagt den übrigen Betreuern mehr über das Paket, als nur eine Beschreibungszeile und der übliche Eintrag im Änderungsprotokoll »Initial release«, der an debian-devel-changes@lists.debian.org gesandt wird.
- Es ist hilfreich für Leute, die von `unstable` zehren (und die die vorderste Frontlinie von Testern bilden). Diese Leute sollten ermutigt werden.

- Die Ankündigungen geben Betreuern und anderen interessierten Parteien ein besseres Gefühl dafür, was gerade geschieht und was im Projekt neu ist.

Bitte lesen Sie unter http://ftp-master.debian.org/REJECT-FAQ.html die häufigsten Gründe für die Ablehnung neuer Pakete.

5.2 Änderungen im Paket aufzeichnen

Von Ihnen vorgenommene Änderungen müssen in debian/changelog aufgezeichnet werden. Diese Änderungen sollten eine kurzgefasste Beschreibung bereitstellen, was geändert wurde, warum (falls dies zweifelhaft ist) und vermerken, ob irgendwelche Fehler geschlossen wurden. Sie zeichnen außerdem auf, wenn das Paket vervollständigt wurde. Diese Datei wird in /usr/share/doc/*Paket*/changelog.Debian.gz oder /usr/share/doc/*Paket*/changelog.gz für native Pakete installiert.

Die Datei debian/changelog entspricht einer bestimmten Struktur mit einer Anzahl unterschiedlicher Felder. Ein bedeutendes Feld, die distribution, wird in Abschnitt 5.5 beschrieben. Weitere Informationen über die Struktur dieser Datei können im Abschnitt debian/changelog der Debian-Richtlinien gefunden werden.

Wenn das Paket im Archiv installiert ist, können Einträge im Änderungsprotokoll benutzt werden, um Debian-Fehler automatisch zu schließen. Siehe Abschnitt 5.8.4.

Es ist üblich, dass der Änderungsprotokolleintrag eines Pakets, der eine neue Originalversion der Software enthält, wie folgt aussieht:

```
  * New upstream release.
```

Es gibt Werkzeuge, die Ihnen helfen Einträge zu erstellen und das changelog zur Veröffentlichung fertigzustellen – siehe Abschnitt A.6.1 und Abschnitt A.6.6.

Siehe auch Abschnitt 6.3.

5.3 Das Paket testen

Bevor Sie Ihr Paket hochladen, sollten Sie es grundlegenden Tests unterziehen. Zumindest sollten Sie die folgenden Dinge ausprobieren (Sie sollten eine ältere Version des gleichen Debian-Pakets zur Hand haben):

- Installieren Sie das Paket und stellen Sie sicher, dass die Software funktioniert oder führen Sie ein Upgrade von der älteren Version auf Ihre neue Version durch, falls bereits ein Debian-Paket existiert.
- Führen Sie für das Paket **lintian** aus. Sie können **lintian** wie folgt ausführen: lintian -v *Paket-Version*.changes. Falls Sie die von **lintian** erzeugte Ausgabe nicht verstehen, versuchen Sie den Schalter **-i** hinzuzufügen. Er veranlasst **lintian** eine viel detailliertere Beschreibung des Problems auszugeben.

  Normalerweise sollte ein Paket *nicht* hochgeladen werden, wenn es **lintian**-Fehler verursacht (sie beginnen mit E).

  Weitere Informationen über **lintian** finden Sie unter Abschnitt A.2.1.
- Führen Sie wahlweise **debdiff** (siehe Abschnitt A.2.2) aus, um Änderungen von einer älteren Version zu analysieren, falls es solche gibt.
- Downgraden Sie das Paket auf die vorherige Version (falls es eine gibt) – dies testet die Skripte postrm und prerm.
- Entfernen Sie das Paket und installieren Sie es erneut.
- Kopieren Sie das Quellpaket in ein anderes Verzeichnis und versuchen Sie es zu entpacken und neu zu erstellen. Dies testet, ob das Paket auf bestehenden Dateien von außen beruht oder ob es auf Benutzerrechten beruht, die in den Dateien konserviert wurden, die innerhalb der .diff.gz-Datei mitgeliefert wurden.

5.4 Layout des Quellpakets

Es gibt zwei Typen von Debian-Quellpaketen:

- die sogenannten native-Pakete, bei denen es keine Unterschiede zwischen den Originalquellen und den auf Debian angewandten Patches gibt

- die (häufigeren) Pakete, bei denen die Original-Quellcode-Tarball-Datei mit einer anderen Datei mitgeliefert wird, die die von Debian vorgenommenen Änderungen enthält

Bei nativen Paketen enthält der Quell-Tarball eine Steuerungsdatei für Debian-Quellen (`.dsc`) und einen Quell-Tarball (`.tar.{gz,bz2,xz}`). Ein Quellpaket eines nicht nativen Paketes enthält eine Steuerungsdatei für Debian-Quellen, den Original-Quellcode-Tarball (`.orig.tar.{gz,bz2,xz}`) und die Debian-Änderungen (`.diff.gz` für das Quellformat »1.0« oder `.debian.tar.{gz,bz2,xz}` für das Quellformat »3.0 (quilt)«).

Mit dem Quellformat »1.0« wurde zur Zeit des Erstellens durch **dpkg-source** festgelegt, ob ein Paket nativ ist oder nicht. Heutzutage wird empfohlen, das gewünschte Quellformat explizit durch Angabe von »3.0 (quilt)« oder »3.0 (native)« in `debian/source/format` festzulegen. Der Rest dieses Abschnitts bezieht sich auf nicht native Pakete.

Anfangs, wenn eine Version hochgeladen wird, die einer bestimmten Version der Ursprungsquelle entspricht, könnte die Original-Tar-Quelldatei hochgeladen und in die `.changes`-Datei eingefügt werden. Nachfolgend sollte eben diese Tar-Datei benutzt werden, um neue .diff- und `.dsc`-Dateien zu erstellen ohne der Notwendigkeit sie erneut hochzuladen.

Standardmäßig werden **dpkg-genchanges** und **dpkg-buildpackage** die Original-Tar-Quelldatei nur dann einbeziehen, falls der aktuelle Änderungsprotokolleintrag eine andere Originalversion des vorhergehenden Eintrags hat. Diese Verhalten könnte durch die Benutzung von `-sa` geändert werden, um es immer einzubeziehen oder `-sd`, um es immer wegzulassen.

Falls im Upload keine Originalquelle enthalten ist, wird die Original-Tar-Quelldatei von **dpkg-source** benutzt, wenn die `.dsc`-Datei erzeugt wird und die Diff-Datei, die hochgeladen werden soll, *muss* Byte für Byte identisch mit der sein, die bereits im Archiv ist.

Bitte beachten Sie, dass in nicht nativen Paketen Zugriffsrechte von Dateien, die nicht in den `*.orig.tar.{gz,bz2,xz}`-Dateien enthalten sind, nicht aufbewahrt werden, da das Diff die Dateizugriffsrechte nicht im Patch speichert. Wenn Sie jedoch das Format «3.0 (quilt)« benutzen, werden Zugriffsrechte von Dateien innerhalb des `debian`-Verzeichnisses aufbewahrt, da sie in einem Tar-Archiv gespeichert werden.

5.5 Eine Distribution herausgreifen

Bei jedem Upload muss angegeben werden, für welche Distribution das Paket gedacht ist. Der Prozess der Paketerstellung extrahiert diese Information aus der ersten Zeile der Datei `debian/changelog` und platziert sie im Feld `Distribution` der `.changes`-Datei.

Es gibt mehrere mögliche Werte für dieses Feld: `stable`, `unstable`, `testing-proposed-updates` und `experimental`. Normalerweise werden Pakete nach `unstable` hochgeladen.

Tatsächlich gibt es zwei andere mögliche Distributionen: `stable-security` und `testing-security`, aber lesen Sie Abschnitt 5.8.5, um weiter Informationen darüber zu erhalten.

Es ist nicht möglich ein Paket gleichzeitig in mehrere Distributionen hochzuladen.

5.5.1 Ein Sonderfall sind Uploads in die Distributionen `stable` und `oldstable`.

Hochladen nach `stable` bedeutet, dass das Paket in die Warteschlange `proposed-updates-new` übertragen wird, damit es von den Release-Verwaltern von Stable überprüft wird. Falls es zugelassen wird, wird es im Verzeichnis `stable-proposed-updates` des Debian-Archivs installiert. Von dort wird es zum nächsten Veröffentlichungszeitpunkt in `stable` eingefügt.

Um sicherzustellen, dass Ihr Upload akzeptiert wird, sollten Sie die Änderungen mit dem Stable-Release-Team besprechen bevor Sie es hochladen. Dazu reichen Sie mittels **reportbug** einen Fehlerbericht gegen das Pseudo-Paket `release.debian.org` ein, einschließlich dem Patch, das Sie auf die derzeit in `stable` enthaltene Paketversion anwenden möchten. Schreiben sie beim Upload in die Distribution `stable` stets detaillierte und ausführliche Änderungsprotokolleinträge.

Sie sollten beim Hochladen nach `stable` besonders vorsichtig sein. Grundsätzlich sollte ein Paket nur nach `stable` hochgeladen werden, wenn folgendes geschieht:

- ein wirklich kritisches Funktionalitätsproblem
- Das Paket ist wird uninstallierbar.
- Einer veröffentlichten Architektur fehlt das Paket.

Früher wurden Uploads nach `stable` benutzt, um auch Sicherheitsprobleme anzugehen. Diese Praxis ist jedoch missbilligt, da Uploads für Debian-Sicherheitswarnungen automatisch in das entsprechende Archiv `proposed-updates` kopiert werden, wenn die Warnung veröffentlicht wird. Weitere detailliertere Informationen über die Handhabung von Sicherheitsproblemem finden Sie unter Abschnitt 5.8.5. Falls das Sicherheits-Team das Problem als zu harmlos erachtet, um es durch ein `DSA` zu beheben. sind die Release-Verwalter nichtsdestotrotz bereit, Ihre Fehlerbehebung bei einem regulären Upload nach `stable` einzufügen.

Es wird abgeraten, etwas anderes unwichtiges im Paket zu ändern, da sogar einfache Fehlerbehebungen zu weiteren Fehlern führen können.

Pakete, die nach `stable` hochgeladen werden, müssen auf Systemen compiliert werden, auf denen `stable` läuft, so dass sich ihre Abhängigkeiten auf die Bibliotheken (und anderen Pakete) beschränken, die auf `stable` verfügbar sind. Ein Paket, das zum Beispiel nach `stable` hochgeladen wurde, das von einem Bibliothekspaket abhängt, das nur in `unstable` existiert, wird zurückgewiesen. Von Änderungen an Abhängigkeiten von anderen Paketen (durch Murksen mit `Provides`- oder `shlibs`-Dateien), die diese anderen Pakete möglicherweise uninstallierbar machen, wird dringend abgeraten.

Uploads nach `oldstable`-Distributionen sind möglich, solange sie nicht archiviert sind. Es gelten die gleichen Regekn, wie für `stable`.

5.5.2 Ein Sonderfall sind Uploads nach `testing/testing-proposed-updates`

Bitte lesen Sie die Informationen im Bereich Testing, um weitere Einzelheiten zu erfahren.

5.6 Ein Paket hochladen

5.6.1 Hochladen auf `ftp-master`

Um ein Paket hochzuladen, sollten Sie die Dateien (einschließlich der signierten Änderungen an der dsc-Datei) mit anonymem FTP nach `ftp.upload.debian.org` in das Verzeichnis /pub/UploadQueue/ hochladen. Damit die Dateien dort verarbeitet werden, benötigen Sie einen signierten Schlüssel im Debian-Entwickler-Schlüsselbund (siehe `http://wiki.debian.org/DebianMaintainer`).

Bitte beachten Sie, dass Sie die Datei »changes« zuletzt übertragen sollten. Andernfalls könnte Ihr Upload abgelehnt werden, da die Archivverwaltungs-Software die »changes«-Datei auswertet und feststellt, dass nicht alle Dateien hochgeladen wurden.

Vielleicht finden Sie auch die Debian-Pakete dupload oder dput nützlich, wenn Sie Pakete hochladen. Diese praktischen Programme helfen den Prozess des Hochladens von Paketen nach Debian zu automatisieren.

Um Pakete zu entfernen, sehen Sie sich bitte `ftp://ftp.upload.debian.org/pub/UploadQueue/README` und das Debian-Paket dcut an.

5.6.2 Verzögerte Uploads

Manchmal ist es nützlich ein Paket sofort hochzuladen, aber zu wünschen, dass dieses Paket das Archiv erst ein paar Tage später erreicht. Sie könnten, wenn Sie beispielsweise einen Non-Maintainer Upload vorbereiten, dem Betreuer ein paar Tage Zeit geben wollen, damit er reagieren kann.

Ein Upload des Pakets in das Verzögerungsverzeichnis, wird es in der deferred uploads queue halten. Wenn die angegebene Wartezeit vorüber ist, wird das Paket zur Verarbeitung in das reguläre Eingangsverzeichnis verschoben. Dies wird durch automatisches Hochladen nach `ftp.upload.debian.org` in das Upload-Verzeichnis `DELAYED/`*`X`*`-day` (*`X`* zwischen 0 und 15). 0-day wird mehrmals täglich nach `ftp.upload.debian.org` hochgeladen.

Mit Dput können Sie den Parameter `--delayed` *`VERZÖGERUNG`* benutzen, um das Paket in eine der Warteschlangen einzureihen.

5.6.3 Sicherheits-Uploads

Laden Sie **KEIN** Paket in die Sicherheits-Upload-Warteschlange (`oldstable-security`, `stable-security` etc.) hoch ohne vorher eine Erlaubnis vom Sicherheits-Team erhalten zu haben. Falls das Paket nicht exakt den Anforderungen des Teams entspricht, wird es viele Probleme und Verzögerungen in der Behandlung des unerwünschten Uploads verursachen. Um Einzelheiten zu erhalten, lesen Sie Abschnitt 5.8.5.

5.6.4 Andere Upload-Warteschlangen

Es gibt in Europa eine alternative Upload-Warteschlange unter `ftp://ftp.eu.upload.debian.org/pub/UploadQueue/`. Sie arbeitet auf die gleiche Weise wie `ftp.upload.debian.org`, sollte aber für europäische Entwickler schneller sein.

Pakete können auch per SSH nach `ssh.upload.debian.org` hochgeladen werden. Dateien sollten in `/srv/upload.debian.org/UploadQueue` abgelegt werden. Diese Warteschlange unterstützt keine verzögerten Uploads.

5.6.5 Benachrichtigung, dass eine neues Paket instaliert wurde

Die Debian-Archivbetreuer sind für die Behandlung der Paket-Uploads verantwortlich. Zum größten Teil werden Uploads automatisch täglich durch das Archiv-Verwaltungswerkzeug **dak process-upload** gehandhabt. Im Besonderen werden Aktualisierungen zu existierenden Paketen in der Distribution `unstable` automatisch eingepfelgt. In anderen Fällen, insbesondere bei neuen Paketen, wird das hochgeladene Paket manuell in die Distribution platziert. Wenn Uploads manuell behandelt werden, kann es einige Zeit dauern bis die Änderung im Archiv erscheint. Bitte haben Sie Geduld.

Auf jeden Fall werden Sie eine E-Mail-Benachrichtigung erhalten, die anzeigt, dass das Paket dem Archiv hinzugefügt wurde und welche Fehler durch den Upload geschlossen werden. Bitte lesen Sie diese Benachrichtigung sorgfältig und prüfen Sie, ob irgendwelche Fehler, die Sie schließen wollten, nicht berücksichtigt wurden.

Die Installationsbenachrichtigung enthält außerdem die Information, in welchen Bereich das Paket eingefügt wird. Falls es dort einen Unterschied gibt, werden Sie eine separate E-Mail-Benachrichtigung darüber erhalten. Lesen Sie das Folgende.

Beachten Sie, dass, falls Sie mittels Warteschlangen hochladen, die Warteschlangen-Daemon-Sortware Ihnen auch per E-Mail Benachrichtigungen sendet.

5.7 Angabe des Paketbereichs, des Unterbereichs und der Priorität

Die Felder `Section` und `Priority` der Datei `debian/control` geben weder an, wo die Datei im Archiv tatsächlich platziert wird noch deren Priorität. Um die gesamte Integrität des Archivs zu wahren, haben die Archivbetreuer die Kontrolle über diese Felder. Die Werte in der Datei `debian/control` sind tatsächlich nur Hinweise.

Die Archivbetreuer behalten den Überblick über die vorschriftsmäßigen Bereiche und Prioritäten für Pakete im `override file`. Falls es dort einen Unterschied zwischen dem `override file` und den Paketfeldern, die in `debian/control` angezeigt werden, gibt, werden Sie eine E-Mail-Benachrichtigung über die Abweichung erhalten, wenn das Paket in das Archiv installiert wird. Sie können entweder Ihre `debian/control`-Datei für Ihren nächsten Upload ändern oder eine Änderung am `override file` wünschen.

Um den tatsächlichen Bereich abzuändern, in den Ihr Paket abgelegt wird, müssen Sie zuerst sicherstellen, dass die Datei `debian/control` in Ihrem Paket fehlerfrei ist. Als nächstes versenden Sie einen Fehlerbericht gegen `ftp.debian.org` mit der Bitte, den Bereich oder die Priorität für Ihr Paket von dem alten auf den neuen Bereich oder die neue Priorität zu ändern. Benutzen Sie einen Betreff wie `override:PACKAGE1:section/priority, [...], PACKAGEX:section/priority` und fügen Sie die Begründung der Änderung in den Nachrichtentext des Fehlerberichts ein.

Weitere Informationen über `override files` finden Sie unter dpkg-scanpackages(1) und `http://www.debian.org/Bugs/Developer#maintincorrect`.

Beachten Sie, dass das Feld `Section` sowohl den Bereich als auch den Unterbereich beschreibt, die in Abschnitt 4.6.1 erläutert werden. Falls der Bereich »main« ist, sollte er weggelassen werden. Die Liste der erlaubten Unterbereiche kann unter `http://www.debian.org/doc/debian-policy/ch-archive.html#s-subsections` gefunden werden.

5.8 Fehlerbehandlung

Jeder Entwickler muss in der Lage sein, mit der Debian-Fehlerdatenbank zu arbeiten. Dies umfasst das Wissen, wie Fehlerberichte richtig eingeordnet werden (siehe Abschnitt 7.1), wie sie aktualisiert und neu geordnet werden und wie sie verarbeitet und geschlossen werden.

Die Funktionen des Fehlerverfolgungssystems sind in unter Fehlerverwaltungssystem für Paket-Betreuer beschrieben. Dies umfasst das Schließen von Fehlern, Followup-Nachrichten, Zuweisen von Schweregraden, Markieren von Fehlern als weitergeleitet und andere Themen.

Operationen, wie das erneute Zuweisen von Fehlern an andere Pakete, das Zusammenführen separater Fehlerberichte zum gleichen Thema oder das Wiedereröffnen von Fehlern, wenn diese voreilig geschlossen wurden, werden vom sogenannten Steuermailserver gehandhabt. All die Befehle, die auf diesem Server verfügbar sind, werden in der Einführung in den E-Mail-Server für die Kontrolle und Manipulation beschrieben.

5.8.1 Fehlerüberwachung

Falls Sie ein guter Paketbetreuer sein möchten, sollten Sie regelmäßig die Debian-Fehlerdatenbank (BTS) für Ihre Pakete überprüfen. Das BTS enthält alle offenen Fehler Ihrer Pakete. Sie können sie prüfen, indem Sie diese Seite durchstöbern: `http://bugs.debian.org/`*`Ihre_Anmeldung`*`@debian.org`.

Paketbetreuer interagieren mit dem BTS über E-Mail-Adressen auf `bugs.debian.org`. Dokumentationen über verfügbare Befehle können Sie unter `http://www.debian.org/Bugs/` finden oder, falls Sie das Paket `doc-debian` installiert haben, können Sie in die lokalen Dateien `/usr/share/doc/debian/bug-*` sehen.

Einige finden es nützlich regelmäßig Berichte über offene Fehler zu erhalten. Sie können einen Cron-Job wie den folgenden hinzufügen, falls Sie wöchentlich eine E-Mail erhalten möchten, die alle Fehler Ihrer Pakete umreißt:

```
# Anfrage wöchentlicher Berichte über Fehler in eigenen Paketen
0 17 * * fri   echo "index maint address" | mail request@bugs.debian.org
```

Ersetzen Sie *Adresse* durch Ihre offizielle Debian-Betreueradresse.

5.8.2 Auf Fehler antworten

Wenn Sie auf Fehler antworten, stellen Sie sicher, dass jegliche Diskussion, die Sie über Fehler führen, sowohl an den Originalabsender, als auch an den Fehler selbst geschickt wird (z.B. *123*@bugs.debian.org). Falls Sie eine neue E-Mail schreiben und sich nicht an die Absender-E-Mail-Adresse erinnern, können Sie die E-Mail-Adresse *123*-submitter@bugs.debian.org benutzen, um den Absender zu kontaktieren *und* Ihre E-Mail innerhalb des Fehlerprotokolls aufzuzeichnen (das bedeuted, dass Sie keine Kopie dieser E-Mail an *123*@bugs.debian.org senden müssen.

Falls Sie einen Fehlerbericht erhalten, der FTBFS erwähnt, so bedeutet dies »Fails to build from source« (Kann nicht aus der Quelle erstellt werden). Portierer benutzen diese Abkürzung öfters.

Sobald Sie einen Fehlerbericht erledigt haben (z.B. den Fehler beheben), markieren Sie ihn als `done` (dies schließt ihn), indem Sie eine Erklärung an *123*-done@bugs.debian.org senden. Falls Sie einen Fehler durch Ändern und Hochladen des Pakets schließen, können Sie das Schließen von Fehlern, wie in Abschnitt 5.8.4 beschrieben, automatisieren.

Sie sollten Fehler *niemals* durch Senden des Befehls `close` an control@bugs.debian.org schließen. Falls Sie dies tun, wird der ursprüngliche Absender keine Informationen darüber erhalten, warum der Fehler geschlossen wurde.

5.8.3 Fehlerorganisation

Als Paketbetreuer werden Sie öfters Fehler in anderen Paketen finden oder Fehlerberichte gegen Ihre Pakete erhalten, die tatsächlich Fehler in anderen Paketen sind. Die Funktionen der Fehlerdatenbank werden in den Informationen über das Fehlerverwaltungssystem für Paket-Betreuer beschrieben. Operationen wie erneutes Zuweisen, Zusammenführen und Markieren von Fehlerberichten werden in der Einführung in den E-Mail-Server für die Kontrolle und Manipulation beschrieben. Dieser Abschnitt enthält einige Richtlinien für die Verwaltung Ihrer eigenen Fehler, die auf der gesammelten Erfahrung der Debian-Entwickler basieren.

Fehler für Probleme einreichen, die Sie in anderen Paketen finden, ist eine der bürgerlichen Pflichten des Betreuerdaseins. Einzelheiten finden Sie unter Abschnitt 7.1. Es ist jedoch wichtiger die Fehler in Ihren eigenen Paketen zu behandeln.

Hier ist eine kurze Liste der Schritte, denen Sie zu Handhabung eines Fehlerberichts folgen können:

1. Entscheiden Sie, ob der Bericht einem echten Fehler entspricht oder nicht. Manchmal rufen Anwender ein Programm nur auf die falsche Art auf, da Sie die Dokumentation nicht gelesen haben. Falls Sie dies diagnostizieren, schließen Sie den Fehler nur und stellen Sie informationen bereits, damit der Anwender sein Problem lösen kann (verweisen Sie auf die gute Dokumentation und so weiter). Falls der gleiche Bericht immer wieder kommt, sollten Sie sich fragen, ob die Dokumentation ausreicht oder ob das Programm den falschen Gebrauch feststellen kann, um eine aussgagekräftige Fehlermeldung auszugeben. Dies ist ein Thema, das Sie mit dem Originalautor angehen sollten.

Falls der Absender des Fehler nicht mit Ihrer Entscheidung, den Fehler zu schließen, einverstanden ist, können Sie den Fehler neu öffnen, bis Sie eine Vereinbarung gefunden haben, wie er gehandhabt werden soll. Falls Sie keine finden, können Sie den Fehler mit `wontfix` markieren, damit die Leute wissen, dass der Fehler existiert, Sie ihn aber nicht beheben möchten. Falls diese Situation nicht akzeptabel ist, können Sie (oder der Absender) eine Entscheidung des technischen Ausschusses anfordern, indem Sie den Fehler neu an `tech-ctte` zuweisen (Sie könnten den Befehl »clone« des BTS verwenden, falls Sie wünschen, dass der Fehlerbericht gegen Ihr Paket weiterbesteht). Lesen Sie, bevor Sie dies tun, die empfohlene Prozedur.

2. Falls der Fehler zwar echt ist, jedoch ein anderes Paket betrifft, weisen Sie ihn nur dem richtigen Paket neu zu. Falls Sie nicht wissen, an welches Paket er zugewiesen werden soll, sollten Sie im IRC oder auf debian-devel@lists.debian.org nach Hilfe fragen. Bitte informieren Sie den/die Paketbetreuer des Pakets, dem Sie den Fehler zuweisen, zum Beispiel indem Sie eine Kopie der E-Mail senden, die das erneute Zuweisen an *Paketname*@packages.debian.org vornimmt und Ihre Beweggründe für diese E-Mail erklären. Bitte beachten Sie, dass ein einfaches erneutes Zuweisen *nicht* an den Betreuer des Pakets versandt wird, an das zugewiesen wird, so dass er nichts davon erfährt, bis er in die Fehlerübersicht seiner Pakete schaut.

   Falls der Fehler die Arbeit Ihres Pakets beeinflusst, denken Sie bitte daran, den Fehler zu klonen und den Klon dem Paket neu zuzuweisen, das das Verhalten tatsächlich verursacht. Andernfalls wird der Fehler nicht in Ihrer Liste der Paketfehler aufgeführt, was Anwender dazu veranlasst, den gleichen Fehler immer wieder zu melden. Sie sollten »Ihren« Fehler mit dem neu zugewiesenen, geklonten Fehler blockieren, um die Beziehung zu dokumentieren.

3. Manchmal müssen Sie außerdem den Schweregrad des Fehlers anpassen, so dass er der Debian-Definition entspricht. Dies geschieht deshalb, weil Leute die Schwere der Fehler aufblähen, um sicherzustellen, dass ihre Fehler rasch behoben werden. Einige Fehler können sogar auf den Schweregrad »wishlist« abgesenkt werden, wenn die angefragte Änderung nur kosmetischer Natur ist.

4. Falls der Fehler echt ist, das gleiche Problem aber bereits von jemand anderem gemeldet wurde, dann sollten die beiden relevanten Fehler mit dem Befehl »merge» des BTS zu einem zusammengefügt werden.Auf diese Art werden alle Absender des Fehlers informiert, wenn er behoben wurde. (Beachten Sie jedoch, dass E-Mails, die an den Absender eines Fehlerberichts gesandt werden, nicht automatisch an alle anderen Absender von Berichten gesandt werden.) Weitere Details über die Form des »merge«-Befehls und dem verwandten Befehl »unmerge« finden Sie in der Dokumentation des BTS-Steuerungs-Servers.

5. Der Absender des Fehlerberichts könnte vergessen haben, einige Informationen bereitzustellen. In diesem Fall müssen Sie die benötigten Informationen bei ihm erfragen. Sie könnten die Kennzeichnung `moreinfo` benutzen, um den Fehler so zu markieren. Außerdem können Sie den Fehler, falls sie ihn nicht reproduzieren können, als `unreproducible` kennzeichnen. Jeder, der den Fehler reproduzieren kann, ist dann eingeladen, weitere Informationen bereitzustellen, wie er reproduziert werden kann. Nach ein paar Monaten kann der Fehler geschlossen werden, falls diese Information von niemandem gesandt wird.

6. Falls sich der Fehler auf die Paketierung bezieht, beheben Sie ihn nur. Falls Sie ihn nicht selbst beheben können, kennzeichnen Sie den Fehler mit `help`. Sie können außerdem auf debian-devel@lists.debian.org oder debian-qa@lists.debian.org um Hilfe ersuchen. Falls es ein Problem der Originalautoren ist, müssen Sie es an die Originalautoren weiterleiten. Es reicht nicht aus, den Fehler nur weiterzuleiten, Sie müssen bei jeder Veröffentlichung prüfen, ob der Fehler behoben wurde oder nicht. Falls dies der Fall ist, schließen Sie ihn, andernfalls müssen Sie den Autor später daran erinnern. Falls Sie über die erforderlichen Fähigkeiten verfügen, können Sie einen Patch vorbereiten, der den Fehler behebt und ihn dem Autor mitschicken. Stellen Sie sicher, dass Sie den Patch an das BTS senden und mit `patch` kennzeichnen.

7. Falls Sie einen Fehler in Ihrer lokalen Kopie behoben haben oder eine Fehlerbehebung in das VCS-Depot übertragen wird, können Sie den Fehler als `pending` kennzeichnen, um die Leute wissen zu lassen, dass der Fehler behoben ist und mit dem nächsten Upload geschlossen wird (dem `changelog` wird `closes:` hinzugefügt). Dies ist besonders nützlich, falls Sie zusammen mit mehreren Entwicklern am Paket arbeiten.

8. Sobald ein korrigiertes Paket im Archiv verfügbar ist, sollte der Fehler geschlossen und dei Version, in der er behoben wurde, angegeben werden. Dies kann automatisch geschehen – lesen Sie Abschnitt 5.8.4.

5.8.4 Wann Fehler durch neue Uploads geschlossen werden

Da Fehler und Probleme in Ihren Paketen behoben werden, liegt es in Ihrer Verantwortung als Paketbetreuer, diese Fehler zu schließen. Sie sollten jedoch keinen Fehler schließen, bis das Paket, das den Fehler schließt, im Debian-

Archiv akzeptiert wurde. Daher können und sollen Sie, sobald Sie die Benachrichtigung erhalten, dass Ihr aktualisiertes Paket in das Archiv installiert wurde, den Fehler im BTS schließen. Außerdem sollte der Fehler mit der korrekten Version geschlossen werden.

Es ist jedoch möglich, das manuelle Schließen von Fehlern nach dem Upload zu vermeiden – führen Sie die behobenen Fehler in Ihrer `debian/changelog`-Datei auf. Folgen Sie dabei einer bestimmten Syntax, dann wird die Verwaltungssoftware die Fehler für Sie schließen. Zum Beispiel:

```
acme-cannon (3.1415) unstable; urgency=low

  * Frobbed with options (closes: Bug#98339)
  * Added safety to prevent operator dismemberment, closes: bug#98765,
    bug#98713, #98714.
  * Added man page. Closes: #98725.
```

Technisch gesehen beschreibt der folgende reguläre Perl-Ausdruck, wie das Schließen von Fehlern in Änderungsprotokollen identifiziert wird:

```
/closes:\s*(?:bug)?\#\s*\d+(?:,\s*(?:bug)?\#\s*\d+)*/ig
```

Die Syntax `closes:#`*XXX* wird bevorzugt, da sie die kürzeste Art des Eintrags ist und am einfachsten in dem Text von `changelog` integriert werden kann. Falls nicht durch den Schalter `-v` von **dpkg-buildpackage** etwas anderes angegeben wurde, werden nur die Fehler im aktuellsten Eintrag des Änderungsprotokolls geschlossen (grundsätzlich werden exakt die Fehler geschlossen, die in der Datei `.changes` im »changelog-part« genannt werden).

Früher wurden Uploads, die als Non-Maintainer Upload (NMU) erkannt wurden, als `fixed` statt als »closed« gekennzeichnet, aber diese Praxis wurde mit dem Beginn der Versionsverfolgung eingestellt. Das gleiche wurde mit der Markierung `fixed-in-experimental` getan.

Falls Sie sich bei der Fehlernummer vertippt haben oder einen Fehler in den Änderungsprotokolleinträgen vergessen haben, zögern Sie nicht, jeglichen durch den Fehler verursachten Schaden rückgängig zu machen. Um fälschlicherweise geschlossene Fehler neu zu öffnen, senden Sie den Befehl `reopen` *XXX* an die Steuerungsadresse control@bugs.debian.org der Fehlerdatenbank. Um irgendwelche verbleibenden Fehler zu schließen, die durch Ihren Upload behoben wurden, mailen Sie die Datei `.changes` an *XXX*-done@bugs.debian.org, wobei *XXX* die Fehlernummer ist und tragen Sie Version: *YYY* und eine leere Zeile als erste zwei Zeilen in den Textteil der Mail ein, wobei *YYY* die erste Version ist, in der der Fehler behoben wurde.

Behalten Sie im Gedächnis, dass es nicht verpflichend ist, Fehler unter Benutzung des Änderungsprotokolls zu schließen, wie oben beschrieben. Falls Sie nur einfach einen Fehler schließen möchten, der mit dem von Ihnen getätigten Upload nichts zu tun hat, können Sie dies durch Mailen einer Erläuterung an *XXX*-done@bugs.debian.org erledigen. Schließen Sie **keine** Fehler im Änderungsprotokolleintrag einer Version, wenn die Änderungen in dieser Version des Pakets keine Bedeutung für den Fehler haben.

Allgemeine Informationen über das Verfassen von Änderungsprotokolleinträgen finden Sie unter Abschnitt 6.3.

5.8.5 Handhabung von sicherheitsrelevanten Fehlern

Aufgrund ihrer sensiblen Natur müssen sicherheitsrelevante Fehler vorsichtig gehandhabt werden. Das Debian-Sicherheits-Team existiert, um diese Aktivitäten zu koordinieren, ausstehende Sicherheitsprobleme zu verfolgen, Paketbetreuern bei Sicherheitsproblemen zu helfen oder sie selbst zu beheben, Sicherheitswarnungen zu senden und `security.debian.org` zu verwalten.

Wenn Sie einen sicherheitsrelevanten Fehler in einem Debian-Paket bemerken, dessen Betreuer Sie sind oder nicht, sammeln Sie sachdienliche Informationen über das Problem und kontaktieren Sie umgehend das Sicherheits-Team, vorzugsweise durch Einreichen eines Eintrags in der Anfragenverfolgung (»Request Tracker«). Siehe `http://wiki.debian.org/rt.debian.org#Security_Team`. Alternativ können Sie eine E-Mail an team@security.debian.org senden. **LADEN SIE KEINE** Pakete für `stable` hoch, ohne das Team zu kontaktieren. Nützliche Informationen enthalten beispielsweise:

- ob der Fehler bereits öffentlich ist oder nicht.
- von welchen Versionen des Pakets bekannt ist, dass sie vom Fehler betroffen sind. Prüfen Sie jede Version, die es in einem unterstützten Debian-Release gibt, ebenso wie `testing` und `unstable`.
- die Art der Fehlerbehebung, falls verfügbar (Patches sind besonders hilfreich)
- jedes reparierte Paket, das Sie für sich selbst vorbereitet haben (senden Sie nur die Dateien `.diff.gz` und `.dsc` und lesen Sie zuerst Abschnitt 5.8.5.4)

- jede Unterstützung, die sie zur Hilfe beim Testen anbieten können (Exploits, Regressionstest etc.)
- jede Information, die für die zur Warnung nötig ist (siehe Abschnitt 5.8.5.3)

Als Betreuer des Pakets sind sie verantwortlich für dessen Verwaltung, sogar im Stable-Release. Sie sind in der besten Position, um Patches zu beurteilen und aktualisierte Pakete zu testen, sehen Sie daher bitte in die folgenden Abschnitte, wie Pakete vorbereitet werden, damit sie vom Sicherheits-Team gehandhabt werden können.

5.8.5.1 Die Sicherheits-Fehlerverfolgung

Das Sicherheits-Team verwaltet eine zentrale Datenbank, die Debian-Sicherheits-Fehlerverfolgung. Diese enthält alle öffentlich verfügbaren Informationen, die über Sicherheitsthemen verfügbar sind: welche Pakete und Versionen betroffen oder repariert sind und ob daher Stable, Testing und/oder Unstable angreifbar sind. Informationen, die immer noch vertraulich sind, werden nicht zur Fehlerverfolgung hinzugefügt.

Sie können Sie nach einem bestimmten Thema durchsuchen, aber auch nach einem Paketnamen. Schauen Sie nach Ihrem Paket, um zu sehen, welche Themen noch offen sind. Bitte stellen Sie, falls Sie können, weitere Informationen über diese Themen bereit oder helfen Sie, sie in Ihrem Paket zu behandeln. Anweisungen finden Sie auf dem Webseiten der Fehlerverfolgung.

5.8.5.2 Vertraulichkeit

Anders als bei den meisten Aktivitäten innerhalb von Debian, werden Informationen über Sicherheitsthemen eine Zeit lang geheim gehalten. Dies erlaubt es Software-Verteibern, ihre Offenlegung zu koordinieren, um die Belastung ihrer Anwender zu minimieren. Ob dies der Fall ist, hängt von der Natur des Problems und der zugehörigen Fehlerbehebung ab und ob bereits eine Angelegenheit an die Öffentlichkeit gesickert ist.

Es gibt mehrere Möglichkeiten, wie Entwickler von einem Sicherheitsproblem erfahren:

- sie bemerken es in einem öffentlichen Forum (Maillingliste, Website etc.)
- jemand verfasst einen Fehlerbericht
- jemand informiert sie per privater E-Mail

In den ersten beiden Fällen ist die Information öffentlich und es ist wichtig, so schnell wie möglich eine Fehlerbehebung zu haben. Im letzen Fall könnte die Information nicht öffentlich sein, In diesem Fall gibt es ein paar mögliche Optionen, mit dem Problem umzugehen:

- Falls die Offenlegung der Sicherheit gering ist, ist es manchmal nicht nötig, das Problem geheim zu halten und es sollte eine Fehlerbehebung erstellt und veröffentlicht werden.
- Falls das Problem ernst ist, sollte diese Information vorzugsweise mit anderen Anbietern geteilt werden, um eine Veröffentlichung zu koordinieren. Das Sicherheits-Team hält Kontakt zu verschiedenen Organisationen und Einzelpersonen, die sich darum kümmern.

Wenn die Person, die das Problem meldet, bittet, dass es nicht offengelegt wird, sollte diese Anfrage auf alle Fälle mit der einleuchtenden Ausnahme gewürdigt werden, das Sicherheits-Team zu informieren, damit der Fehler für ein Stable-Release von Debian erstellt werden kann. Vergessen Sie nicht, diese Tatsache zu erwähnen, wenn vertrauliche Informationen zum Sicherheits-Team gesandt werden.

Bitte beachten Sie, dass Sie keine Fehlerbehebung nach `unstable` (oder an eine andere Stelle, wie ein öffentliches VCS-Depot) senden können, wenn Geheimhaltung nötig ist. Es reicht nicht aus, die Einzelheiten der Änderung zu verschleiern, da der Code selbst öffentlich ist und von der Allgemeinheit untersucht werden kann (und soll).

Es gibt zwei Gründe, Informationen sogar dann zu veröffentlichen, wenn um Geheimhaltung gebeten wurde: Das Problem ist bereits seit einer Weile bekannt oder es wurde ein Exploit veröffentlicht.

Das Sicherheits-Team hat einen PGP-Schlüssel, um verschlüsselte Kommunikation über sensible Themen zu aktivieren. Einzelheiten finden Sie in der Debian Sicherheits-FAQ.

5.8.5.3 Sicherheitswarnungen

Sicherheitswarnungen werden nur für die aktuelle, veröffentlichte Stable-Distribution ausgegeben und *nicht* für `testing` oder `unstable`. Wenn Sie veröffentlicht werden, werden sie an die Mailingliste debian-security-announce@lists.debian.org und an die Website Sicherheits-Informationen geschickt. Sicherheitswarnungen werden vom Sicherheits-Team verfasst und verschickt. Es macht natürlich nichts aus, wenn ein Paketbetreuer einige Informationen dazu bereitstellen kann oder einen Teil des Textes verfasst. Informationen in einer Warnung sollten Folgendes umfassen:

- eine Beschreibung des Problems und seines Geltungsbereichs, einschließlich:
  - dem Typ des Problems (Rechteausweitung, Dienstverweigerung etc.)
  - Welche Privilegien können erlangt werden und durch wen (falls durch jemanden)?
  - Wie kann dies ausgenutzt werden?
  - Ist es aus der Ferne oder lokal ausnutzbar?
  - Wie wurde das Problem gelöst?

  Diese Informationen ermöglichen es Anwendern die Bedrohung ihrer Systeme zu beurteilen.
- Versionsnummern betroffener Pakete
- Versionsnummern reparierter Pakete
- Informationen, woher man die aktualisierten Pakete bekommen kann (üblicherweise aus dem Debian-Sicherheitsarchiv)
- Referenzen zu Warnungen der Originalautoren, CVE-Bezeichner und jede andere nützliche Informationen in Querverweisen zur Schwachstelle

5.8.5.4 Pakete vorbereiten, um Sicherheitsthemen anzugehen

Eine Möglichkeit, dem Sicherheits-Team bei seinen Aufgaben beizustehen besteht darin, es mit reparierten Paketen zu versorgen, die für eine Sicherheitswarnung des Stable-Releases geeignet sind.

Wenn eine Aktualisierung am Stable-Release vorgenommen wird, muss dies behutsam getan werden, damit eine Änderung des Systemverhaltens vermieden wird und keine neuen Fehler eingeschleppt werden. Um dies zu erreichen, ändern Sie so wenig wie möglich, wenn Sie Fehler beheben. Anwender und Administratoren verlassen sich auf das exakte Verhalten des Release nachdem es veröffentlicht wurde, so dass jegliche vorgenommene Änderung das System von jemandem stören könnte. Dies trifft im Besonderen auf Bibliotheken zu: Stellen Sie sicher, dass Sie nie das API oder das ABI ändern, egal wie klein die Änderung auch sein mag.

Dies bedeutet, dass das Umschwenken auf eine neue Originalversion keine gute Lösung ist. Stattdessen sollten die passenden Änderungen auf die Versionen im aktuellen Debian-Stable-Release zurückportiert werden. Generell sind Original-Paketbetreuer bereit zu helfen, wenn nötig. Falls nicht, könnte das Debian-Sicherheits-Team in der Lage sein zu helfen.

In manchen Fällen ist es unmöglich eine Sicherheitsreparatur zurück zu portieren, zum Beispiel, wenn große Teile des Quelltextes geändert oder überschrieben werden müssen. Falls dies geschieht, könnte es nötig sein, zu einer neue Originalversion zu wechseln. Dies wird jedoch nur in extremen Situationen getan und Sie müssen dies immer vorab mit dem Sicherheits-Team abstimmen.

Darauf bezieht sich eine weitere wichtige Richtlinie: Testen Sie immer Ihre Änderungen. Falls Sie über ein Exploit verfügen, probieren Sie es aus und sehen Sie, ob es wirklich beim nicht reparierten Paket erfolgreich ist und am reparierten Paket scheitert. Testen Sie auch andere normale Aktionen, da eine Sicherheitsreparatur manchmal scheinbar nicht betroffene Funktionen auf raffinierte Weise stört.

Nehmen Sie **KEINE** Änderungen in Ihr Paket auf, die sich nicht direkt auf die Reparatur der Schwachstelle beziehen. Diese müssten rückgängig gemacht werden, was Zeit kostet. Falls es in Ihrem Paket andere Fehler gibt, die Sie gerne beheben würden, machen Sie auf dem üblichen Weg ein Upload nach »proposed-updates«, nachdem die Sicherheitswarnung veröffentlicht wurde. Der Sicherheits-Aktualisierungsmechanismus ist kein Mittel, um Änderungen an Ihrem Paket einzuführen, die andernfalls vom Stable-Release abgelehnt worden wären, unterlassen sie es also.

Überprüfen und testen Sie Ihre Änderungen so ausgiebig wie möglich. Prüfen Sie die Unterschiede zur vorherigen Version mehrmals (**interdiff** aus dem Paket `patchutils` und **debdiff** aus `devscripts` sind nützliche Werkzeuge dafür, siehe Abschnitt A.2.2).

Überprüfen Sie unbedingt folgende Elemente:

- **Visieren Sie** in Ihrem `debian/changelog` die richtige Version an. Für `stable` ist dies `stable-security`, für `testing` ist dies `testing-security` und für das vorherige Stable-Release ist dies `oldstable-security`. Peilen Sie nicht *`Distribution`*`-proposed-updates` oder `stable` an!
- Der Upload sollte die **urgency=high** haben.

- Verfassen Sie anschauliche, aussagekräftige Änderungsprotokolleinträge. Andere werden sich darauf verlassen, um zu bestimmen, ob ein bestimmter Fehler behoben wurde. Fügen Sie für alle eingereichten **Debian-Fehler** `closes:`-Angaben hinzu. Beziehen Sie immer einen externen Bezug ein, vorzugsweise einen **CVE-Bezeichner**, so dass Querverweise darauf möglich sind. Wenn ein CVE-Bezeichner jedoch noch nicht zugewiesen wurde, warten Sie nicht darauf, fahren Sie aber mit dem Prozess fort. Ein späterer Querverweis auf den Bezeichner ist möglich.

- Stellen Sie sicher, dass die **Versionsnummer** angemessen ist. Sie muss größer als die des aktuellen Pakets, aber kleiner als die von Paketversionen in neueren Distributionen sein. Falls es Zweifel gibt, prüfen Sie es mit `dpkg --compare-versions`. Seien Sie vorsichtig, dass Sie keine Versionsnummer wiederverwenden, die Sie für einen vorherigen Upload benutzt haben oder eine, die Konflikte mit einem binNMU auslöst. Es ist Brauch `+Codename1` anzuhängen, z.B. `1:2.4.3-4+lenny1` und natürlich bei nachfolgenden Uploads um eins zu erhöhen.

- Sofern die Originalquelle nicht vorher nach `security.debian.org` hochgeladen wurde, (durch eine vorhergehende Sicherheitsaktualisierung) erstellen Sie den Upload **aus vollständigen Originalquellen** (`dpkg-buildpackage -sa`). Falls es einen vorhergehenden Upload nach `security.debian.org` mit der gleichen Originalversion gab, könnten Sie ohne die Originalquelle hochladen (`dpkg-buildpackage -sd`).

- Tragen Sie Sorge, dass die **exakt gleiche `*.orig.tar.{gz,bz2,xz}`**-Datei, wie im normalen Archiv benutzt wird. Andernfalls ist es nicht möglich, die Sicherheitsfehlerbehebung später in die Hauptarchive zu verschieben.

- Erstellen Sie das Paket auf einem **einwandfreien System**, auf dem nur Pakete der Distribution installiert sind, für die Sie es erstellen. Falls Sie selbst nicht über ein solches System verfügen, können Sie eine debian.org-Maschine verwenden (siehe Abschnitt 4.4) oder richten Sie ein Chroot ein (siehe Abschnitt A.4.3 und Abschnitt A.4.2).

5.8.5.5 Hochladen eines reparierten Pakets

Laden Sie **KEIN** Paket in die Sicherheits-Upload-Warteschlange (`oldstable-security`, `stable-security` etc.) ohne vorherige Genehmigung des Sicherheits-Teams. Falls das Paket nicht exakt den Anforderungen des Teams entspricht, wird es viele Probleme und Verzögerungen im Umgang mit dem unerwünschten Upload verursachen.

Laden Sie ihre Fehlerbehebung **NICHT** nach `proposed-updates` ohne Abstimmung mit dem Sicherheits-Team hoch. Pakete von `security.debian.org` werden automatisch direkt in das Verzeichnis `proposed-updates` kopiert. Falls bereits ein Paket mit der gleichen oder einer höheren Versionsnummer im Archiv installiert ist, wird die Sicherheitsaktualisierung durch das Archivsystem abgelehnt. Stattdessen endet auf diese Weise die Distribution Stable ohne eine Sicherheitsaktualisierung für dieses Paket.

Sobald Sie das neue Paket erstellt und getestet haben und es vom Sicherheits-Team zugelassen wurde, muss es hochgeladen werden, so dass es in den Archiven installiert werden kann. Sicherheits-Uploads werden nach `ftp://security-master.debian.org/pub/SecurityUploadQueue/` hochgeladen.

Sobald ein Upload in die Sicherheitheitswarteschlange akzeptiert wurde, wird das Paket automatisch für alle Architekturen erstellt und zur Überprüfung durch das Sicherheits-Team gespeichert.

Auf Uploads, die auf Zustimmung oder Prüfung warten, kann nur das Sicherheits-Team zugreifen. Dies ist nötig, da es sich um Fehlerbehebungen für Sicherheitsprobleme handeln könnte, die noch nicht offengelegt werden können.

Falls ein Mitglied des Sicherheits-Teams ein Paket akzeptiert, wird es auf `security.debian.org` installiert. Ebenso wird es für das passende *`Distribution`*`-proposed-updates` auf `ftp-master.debian.org` vorgeschlagen.

5.9 Verschieben, Entfernen, Umbenennen, Adoptieren und Verwaisen von Paketen

Einige Operationen zum Manipulieren von Archiven sind im Debian-Upload-Prozess nicht automatisiert. Diesen Prozeduren sollte manuell durch Paketbetreuer gefolgt werden. Dieses Kapitel gibt einen Leitfaden, was in diesen Fällen zu tun ist.

5.9.1 Pakete verschieben

Manchmal ändert ein Paket seinen Bereich. Ein Paket aus dem Bereich `non-free` könnte zum Beispiel in einer neueren Version unter der GPL erscheinen. In diesem Fall sollte es nach »main« oder »contrib« verschoben werden.[1]

Falls Sie für eines Ihrer Pakete den Bereich ändern müssen, ändern Sie die Paketsteuerungsinformation, um das Paket in den gewünschten Bereich zu platzieren und laden Sie das Paket erneut hoch (Einzelheiten finden Sie im Debian Policy Manual). Sie müssen sicherstellen, dass Sie Ihrem Upload die `.orig.tar.{gz,bz2,xz}`-Datei beifügen (sogar, wenn Sie keine neue Originalversion hochladen) sonst es wird nicht zusammen mit dem Rest des Pakets in dem neuen Bereich erscheinen. Falls Ihr neuer Bereich gültig ist, wird es automatisch verschoben. Falls dies nicht geschieht, wenden Sie sich an die Ftpmasters, damit Sie verstehen, was geschehen ist.

Falls Sie andererseits die `subsection` eines Ihrer Pakete ändern müssen (z.B. »devel«, »admin«), ist die Prozedur etwas anders. Korrigieren Sie den in der Steuerungsdatei gefundenen Unterbereich des Pakets und laden Sie es erneut hoch. Außerdem müssen Sie die Datei »override« aktualisieren, wie es in Abschnitt 5.7 beschrieben wird.

5.9.2 Pakete entfernen

Falls Sie aus irgendeinem Grund das Paket vollständig entfernen möchten (etwa, weil es eine alte Kompatibilitätsbibliothek ist, die nicht länger erforderlich ist), müssen sie einen Fehler gegen `ftp.debian.org` einreichen, in dem Sie darum bitten, das Paket zu entfernen. Wie alle Fehler sollte auch dieser normalerweise den Schweregrad »normal« haben. Der Fehlertitel sollte die Form `RM:` *`Paket [Architekturenliste] --Grund`* haben, wobei *`Paket`* der Name des zu entfernenden Pakets und *`Grund`* eine kurze Zusammenfassung sein sollte, aus welchem Grund um Entfernen gebeten wird. *`[Architekturenliste]`* ist optional und wird nur benötigt, wenn das Entfernen lediglich einige Architekturen betrifft, aber nicht alle. Beachten Sie, dass **reportbug** einen Titel erstellt, der diesen Regeln entspricht, wenn Sie es benutzen, um einen Fehler des Pseudo-Pakets `ftp.debian.org` melden.

Falls Sie ein Paket entfernen wollen, das Sie betreuen, sollten Sie dies im Fehlertitel durch Voranstellen von `ROM` (Request Of Maintainer) anmerken. Es gibt mehrere andere Standardabkürzungen, die als Grund für das Entfernen von Paketen benutzt werden. Eine komplette Liste finden Sie unter `http://ftp-master.debian.org/removals.html`. Diese Seite stellt außerdem einen praktischen Überblick über ausstehende Anfragen zum Entfernen bereit.

Beachten Sie, dass Pakete nur aus den Distributionen `unstable`, `experimental` und `stable` entfernt werden können. Pakete werden nicht direkt aus `testing` entfernt. Sie werden vielmehr automatisch entfernt, nachdem das Paket aus `unstable` entfernt wurde und in `testing` kein Paket davon abhängt.

Es gibt eine Ausnahme, bei der eine explizite Anfrage zum Entfernen nicht nötig ist: Falls ein (Quell- oder Binär-) Paket nicht länger aus der Quelle erstellt wird, wird es halbautomatisch entfernt. Bei einem Binärpaket ist dies der Fall, wenn kein Quellpaket dieses Binärpaket weiterhin erzeugt. Falls das Binärpaket nur auf einigen Architekturen nicht länger erstellt wird, ist eine Anfrage zum Entfernen weiterhin nötig. Für ein Quell-Paket bedeutet dies, dass alle Binärpakete, die sich darauf beziehen, von einem anderen Quellpaket übernommen werden müssen.

In Ihrer Bitte um Entfernung müssen Sie detaillierte Gründe angeben, die das Entfernen rechtfertigen. Dies muss so sein, um unerwünschtes Entfernen zu vermeiden und um eine Chronik aufzubewahren, weshalb das Paket entfernt wurde. Sie können zum Beispiel den Namen des Pakets bereitstellen, das das entfernte ersetzt.

Üblicherweise bitten Sie nur ein Paket zu entfernen, das Sie selbst betreuen. Falls Sie ein anderes Paket entfernen möchten, müssen Sie die Genehmigung seines Betreuers einholen. Sollte das Paket verwaist sein und daher keinen Betreuer haben, sollten Sie die Bitte um Entfernung zuerst auf debian-qa@lists.debian.org diskutieren. Falls es dort eine Übereinkunft gibt, dass das Paket entfernt werden soll, sollten Sie den `O:`-Fehler mit einem neuen Titel dem `wnpp`-Paket neu zuweisen, anstatt einen neuen Fehlerbericht als Bitte um Entfernen einzureichen.

Weitere Informationen über diese oder andere Themen, die sich auf das Entfernen von Paketen beziehen, können unter `http://wiki.debian.org/ftpmaster_Removals` und `http://qa.debian.org/howto-remove.html` gefunden werden.

Wenn Zweifel bestehen, ob ein Paket weggeworfen werden kann, fragen Sie per E-Mail an debian-devel@lists.debian.org nach Meinungen. Außerdem ist das Programm **apt-cache** aus dem Paket `apt` von Interesse. Wenn es mit `apt-cache showpkg` *`Paket`* aufgerufen wird, zeigt es Einzelheiten über das *`Paket`*, einschließlich umgekehrter Abhängigkeiten. Andere nützlich Programme umfassen **apt-cache rdepends**, **apt-rdepends**, **build-rdeps** (im Paket `devscripts`) und **grep-dctrl**. Das Entfernen von verwaisten Paketen wird auf debian-qa@lists.debian.org diskutiert.

Sobald das Paket entfernt wurde, sollten die Fehler des Pakets behandelt werden. Sie sollten entweder im Fall, dass der tatsächliche Code in einem anderen Paket entwickelt wurde, neu zugewiesen werden (z.B. `libfoo12` wurde entfernt, weil `libfoo13` es ersetzt) oder geschlossen werden, falls die Software einfach nicht länger Teil

[1] Einen Leitfaden, in welchen Bereich ein Paket gehört, finden Sie im Debian Policy Manual.

von Debian ist. Wenn die Fehler geschlossen werden, sollten sie in der Version `<most-recent-version-ever-in-Debian>+rm` als behoben gekennzeichnet werden, um zu verhindern, dass sie in vorherigen Debian-Releases als behoben gekennzeichnet werden.

5.9.2.1 Entfernen von Paketen aus `Incoming`

Früher war es möglich, Pakete aus `incoming` zu entfernen. Mit der Einführung des neuen Incoming-Systems ist dies jedoch nicht länger möglich. Stattdessen müssen Sie eine neue Überarbeitung Ihres Pakets mit einer höheren Versionsnummer als der des zu ersetzenden Pakets hochladen. Beide Versionen werden im Archiv installiert, aber nur die höhere Version wird tatsächlich in `unstable` verfügbar sein, da die vorherige sofort durch die höhere ersetzt wird. Falls Sie jedoch Ihr Paket ordnungsgemäß testen, sollte es ohnehin nicht allzu oft vorkommen, dass Sie ein Paket ersetzen.

5.9.3 Umbenennen oder Ersetzen von Paketen

Wenn sich die Originalautoren eines Ihrer Pakete entscheiden, ihre Software umzubenennen (oder Ihnen beim Benennen Ihres Pakets ein Fehler unterlaufen ist), sollten Sie einen zweistufigen Prozess durchlaufen, um es umzubenennen. Im ersten Schritt ändern Sie die Datei `debian/control`, damit Sie den neuen Namen wiederspiegelt, ersetzt, bereitzustellt und zu dem veralteten Paketnamen in Konflikt tritt (Einzelheiten finden Sie im Debian Policy Manual). Bitte beachten Sie, dass Sie nur dann eine `Provides`-Beziehung hinzufügen sollten, wenn alle Pakete, die von dem veralteten Paketnamen abhängen, nach dem Umbenennen weiter funktionieren. Sobald Sie das Paket hochgeladen haben und das Paket in das Archiv verschoben wurde, reichen Sie einen Fehler gegen `ftp.debian.org` ein, in dem Sie um das Entfernen des veralteten Namens ersuchen (siehe Abschnitt 5.9.2). Vergessen Sie nicht, gleichzeitig die Fehler ordnungsgemäß neu zuzuweisen.

Sonst könnten Sie einen Fehler beim Konstruieren Ihres Pakets begehen und wünschen, es zu ersetzen. Die einzige Möglichkeit, dies zu tun besteht im Erhöhen der Versionsnummer und dem Hochladen der neuen Version. Die alte Version verliert wie üblich ihre Gültigkeit. Beachten Sie, dass dies auf jeden Teil Ihres Pakets zutrifft, einschließlich der Quellen: Falls Sie den Originalquell-Tarball Ihres Pakets ersetzen möchten, müssen Sie ihn mit einer verschiedenen Version hochladen. Eine einfache Möglichkeit ist es, `foo_1.00.orig.tar.gz` durch `foo_1.00+0.orig.tar.gz` oder `foo_1.00.orig.tar.bz2` zu ersetzen. Diese Einschränkung gibt jeder Datei auf der FTP-Site einen einzigartigen Namen, der dabei hilft, die Einheitlichkeit über ein Netzwerk von Spiegelservern sicherzustellen.

5.9.4 Verwaisen von Paketen

Falls Sie ein Paket nicht länger betreuen können, müssen Sie andere informieren und dafür sorgen, dass das Paket als verwaist gekennzeichnet wird. Sie sollten den Paketbetreuer auf `Debian QA Group <packages@qa.debian.org>` setzen und einen Fehlerbericht gegen das Pseudopaket `wnpp` senden. Der Fehlerbericht sollte mit dem Titel `O:`*`Paket --kurze Beschreibung`* angeben, dass das Paket nun verwaist ist. Der Schweregrad des Fehlers sollte auf `normal` gesetzt werden; falls das Paket die Priotität »standard« oder höher hat, sollte er auf »important« gesetzt werden. Wenn Sie es für nötig halten, senden Sie eine Kopie an debian-devel@lists.debian.org, indem Sie die Adresse in die Kopfzeile X-Debbugs-CC: der Nachricht einfügen (nein, benutzen Sie nicht CC:, da auf diese Art der Betreff der Nachricht die Fehlernummer nicht angibt).

Falls Sie nur die Absicht haben, das Paket abzugeben, aber im Moment noch Betreuer bleiben können, dann sollten Sie stattdessen einen Fehlerbericht gegen `wnpp` mit dem Titel `RFA:`*`Paket --kurze Beschreibung`* senden. `RFA` steht für `Request For Adoption` (Bitte um Adoption).

Weitere Informationen finden Sie auf den WNPP-Web-Seiten.

5.9.5 Adoption eines Pakets

Eine Liste von Paketen, die einen neuen Betreuer suchen, ist unter Arbeit-bedürfende und voraussichtliche Pakete (WNPP) verfügbar. Falls Sie die Verwaltung von einigen Paketen übernehmen möchten, die auf WNPP aufgeführt sind, lesen Sie bitte besagte Seite, um Informationen zu erhalten und etwas über die Prozeduren zu erfahren.

Es ist nicht in Ordnung einfach ein Paket zu übernehmen, das vernachlässigt ist – das wäre Paketentführung. Sie können natürlich den aktuellen Betreuer kontaktieren und ihn fragen, ob Sie das Paket übernehmen dürfen. Falls Sie aus irgend einem Grund annehmen, der Betreuer sei AWOL (absent without leave/abwesend ohne etwas zu hinterlassen), dann lesen Sie Abschnitt 7.4.

Generell sollten Sie das Paket nicht ohne die Zustimmung des aktuellen Betreuers übernehmen. Sogar, wenn er Sie ignoriert, ist das immer noch kein Grund das Paket zu übernehmen. Beschwerden über Betreuer sollten auf der

Entwickler-Mailingliste vorgebracht werden. Falls die Diskussion mit keinem positiven Fazit endet und das Thema technischer Natur ist, erwägen Sie, die Aufmerksamkeit des Technischen Ausschusses darauf zu lenken (weitere Informationen finden Sie unter Debians Technischer Ausschuss).

Wenn Sie ein altes Paket übernehmen, möchten Sie wahrscheinlich als offizieller Betreuer in der Fehlerdatenbank aufgeführt werden. Dies geschieht automatisch, sobald Sie eine neue Version mit einem aktualisierten `Maintainer`-Feld hochladen, obwohl dies nach dem Upload ein paar Tage dauern kann. Falls Sie für eine Weile nicht planen eine neue Version hochzuladen, können Sie das Abschnitt 4.10 benutzen, um Fehlerberichte zu erhalten. Stellen Sie jedoch sicher, dass der alte Betreuer kein Problem damit hat, dass Sie ab diesem Zeitpunkt die Fehlerberichte erhalten.

5.10 Portieren und portiert werden

Debian unterstützt eine immer größer werdende Anzahl von Architekturen. Sogar wenn Sie kein Portierer sind und nur eine einzige Architektur nutzen, gehört es zu Ihren Pflichten als Betreuer die Fragen der Portierbarkeit zu kennen. Daher sollten Sie sogar wenn Sie kein Portierer sind, das meiste in diesem Kapitel lesen.

Portieren ist das Erstellen von Debian-Paketen für Architekturen, die sich von der Originalarchitektur des Binärpakets des Paketbetreuers unterscheiden. Es ist eine einzigartige und notwendige Aktivität. Tatsächlich sind Portierer diejenigen, die meisten Debian-Pakete compilieren. Wenn zum Beispiel ein Paketbetreuer ein (portierbares) Quellpaket mit Binärdateien für die Architektur `i386` hochlädt, wird es für jede andere Architektur erstellt, was auf 11 weitere Builds hinausläuft.

5.10.1 Seien Sie freundlich zu Portierern

Portierer haben schwere und ungewöhnliche Aufgaben, da sie mit einer großen Zahl von Paketen umgehen müssen. Idealerweise sollte jedes Quellpaket richtig aus dem Stand erstellt werden. Unglücklicherweise ist das oft nicht der Fall. Dieser Abschnitt enthält eine Prüfliste von »Patzern«, die öfters von Debian-Betreuern begangen werden – übliche Probleme, die Portierer oft in die Klemme geraten lassen und ihre Arbeit unnötig erschweren.

Die Erste und Wichtigste ist es, schnell auf einen Fehler oder ein Problem zu antworten, das ein Portierer aufgetrieben hat. Behandeln Sie Portierer mit Höflichkeit, da Sie praktisch Mitbetreuer Ihres Pakets sind (was sie gewissermaßen sind). Bitte seien Sie tolerant bei knappen oder sogar unklaren Fehlerberichten. Tun Sie Ihr Bestes, um Jagd auf das zu machen, was auch immer das Problem ist.

Die mit Abstand meisten Probleme, die von Portierern gefunden werden, werden durch *Paketierungsfehler* in den Quellpaketen verursacht. Hier ist eine Prüfliste der Dinge, die Sie prüfen oder wissen sollten.

1. Stellen Sie sicher, dass Ihre `Build-Depends`- und `Build-Depends-Indep`-Einstellungen in der Datei `debian/control` richtig gesetzt sind. Die beste Methode dies zu überprüfen, ist die Benutzung des Pakets `debootstrap`, um eine `unstable`-Chroot-Umgebung zu erstellen (siehe Abschnitt A.4.2). Innerhalb der Chroot-Umgebung installieren Sie das Paket `build-essential` und/oder `Build-Depends-Indep`. Am Ende versuchen Sie Ihr Paket innerhalb der Chroot-Umgebung zu erstellen. Diese Schritte können mit dem Programm **pbuilder** automatisiert werden, das im vom Paket mit dem gleichen Namen bereitgestellt wird (sieheAbschnitt A.4.3).

   Falls Sie kein ordnungsgemäßes Chroot einrichten können, könnte Ihnen **dpkg-depcheck** behilflich sein (siehe Abschnitt A.6.7).

   Anweisungen über die Einrichtung von Erstellungsabhängigkeiten finden Sie im Debian Policy Manual.

2. Setzen Sie »architecture« auf keinen anderen Wert als `all` oder `any`, außer wenn Sie das wirklich beabsichtigen. In zu vielen Fällen folgen Paketbetreuer nicht den Anweisungen im Debian Policy Manual. Wenn Sie Ihre »architecture« nur auf eine Architektur (wie `i386` oder `amd64`) setzen, ist dies normalerweise falsch.

3. Stellen Sie sicher, dass das Quellpaket korrekt ist. Führen Sie `dpkg-source -x` *`Paket`*`.dsc` aus, um sicherzustellen, dass Ihr Quellpaket ordnungsgemäß entpackt wird. Dann versuchen Sie dort hinein Ihr Paket von Grund auf mit **dpkg-buildpackage** zu erstellen.

4. Stellen Sie sicher, dass Sie Ihr Quellpaket nicht mit den Dateien `debian/files` oder `debian/substvars` ausliefern. Sie sollten durch das Target `clean` von `debian/rules` entfernt werden.

5. Stellen Sie sicher, dass Sie sich nicht auf lokal installierte Pakete oder gehackte Konfigurationen oder Programme verlassen. Sie sollten zum Beispiel niemals Programme in `/usr/local/bin` oder dergleichen aufrufen. Versuchen Sie Ihr Paket auf einem anderen Rechner zu erstellen, sogar wenn er die gleiche Architektur hat.

6. Verlassen Sie sich nicht darauf, dass das Paket, das Sie erstellen, bereits installiert ist (ein Teilaspekt des vorherigen Problems). Es gibt natürlich Ausnahmen von dieser Regel, aber seine Sie sich bewusst, dass dies auf jeden Fall manuelles Bootstrapping erfordert und nicht durch automatisierte Paket-Builder erledigt werden kann.

7. Verlassen Sie sich, wenn möglich, nicht auf eine bestimmte Version des Compilers. Falls doch, dann stellen Sie sicher, dass Ihre Build-Abhängigkeiten diese Einschränkungen widerspiegeln, obwohl Sie sich wahrscheinlich Ärger einhandeln, da verschiedene Architekturen manchmal unterschiedliche Compiler vorgeben.

8. Sorgen Sie dafür, dass Ihre `debian/rules`-Datei separate `binary-arch`- und `binary-indep`-Targets enthält, wie es das Debian Policy Manual erfordert. Stellen Sie sicher, dass beide Targets unabhängig voneinander funktionieren, damit Sie ein Target aufrufen können, ohne das Sie vorher das andere aufgerufen haben müssen. Um dies zu prüfen, führen Sie **dpkg-buildpackage -B** aus.

5.10.2 Richtlinien für Uploads von Portierern

Wenn das Paket aus dem Stand für die Architektur erstellt werden kann, auf die es portiert werden soll, haben Sie Glück und Ihre Arbeit ist einfach. Dieser Abschnitt befasst sich mit diesem Fall; er beschreibt, wie Ihr Binärpaket erstellt und hochgeladen wird, so dass es ordnungsgemäß im Archiv installiert werden kann. Falls Sie das Paket patchen müssen, um es für eine andere Architektur compilieren zu können, führen Sie in Wirklichkeit ein Quell-NMU durch, ziehen Sie daher stattdessen Abschnitt 5.11.1 zu Rate.

Für einen Upload eines Portierern werden keine Änderungen an den Quellen vorgenommen. Sie müssen keine Dateien im Quellpaket anfassen. Dies schließt `debian/changelog` ein.

Die Vorgehensweise **dpkg-buildpackage** aufzurufen ist wie folgt: `dpkg-buildpackage -B -m`*`E-Mail des Portierers`*. Natürlich setzen Sie *`E-Mail des Portierers`* auf Ihre E-Mail-Adresse. Dies wird zu einem rein binären Build von nur den Paketteilen führen, die architekturabhängig sind. Dabei wird in `debian/rules` das Target `binary-arch` benutzt.

Falls Sie für Ihr Portierungs-Bestreben auf einer Debian-Maschine arbeiten und Ihren Upload lokal signieren müssen, damit er im Archiv akzeptiert wird, können Sie **debsign** für Ihre `.changes`-Datei ausführen, um sie bequem zu signieren oder benutzen Sie den Signierungsmodus aus der Ferne von **dpkg-sig**.

5.10.2.1 Neu compilieren oder rein binärer NMU

Manchmal ist der erste Upload einer Portierung problematisch, da die Umgebung, in der das Paket erstellt wurde, nicht gut genug war (veraltete oder hinfällige Bibliotheken, falsche Compiler etc.). Dann könnte es nötig sein, dass Sie es nur neu in einer aktualisierten Umgebung compilieren müssen. In diesem Fall müssen Sie jedoch die Versionsnummer ändern, so dass das alte, falsche Paket im Debian-Archiv ersetzt werden kann (**dak** lehnt die Installation neuer Pakete ab, falls Sie keine höheren Versionsnummern, als das aktuell verfügbare haben).

Sie müssen sicherstellen, dass Ihr rein binärer NMU das Paket nicht uninstallierbar macht. Dies könnte geschehen, wenn ein Quellpaket architekturabhängige und architekturunabhängige Pakete generiert, die unter Benutzung von der ersetzbaren Dpkg-Variable `$(Source-Version)` wechselseitige Abhängigkeiten erzeugen.

Ungeachtet der nötigen Modifikation des Änderungsprotokolls, werden diese rein binäre NMUs genannt – es ist nicht nötig in diesem Fall dafür zu sorgen, dass alle anderen Architekturen sich selbst als veraltet oder eines erneuten Compilierens bedürfig betrachten.

Solche Neu-Compilierungen benötigen eine spezielle »magische« Versionsnummerierung, so dass die Archiv-Verwaltungswerkzeuge dies erkennen, selbst wenn es eine neue Debian-Version ist, gibt es dort keine zugehörige Aktualisierung der Quelle. Falls Sie dabei einen Fehler machen, werden die Archivbetreuer Ihre Aktualisierung ablehnen (aus Mangel an entsprechendem Quellcode).

Die »Magie« für ein reines Neu-Compilierungs-NMU wird durch eine Endung ausgelöst, die an die Paketversionsnummer angehängt wird und die Form `b`*`Zahl`* hat. Wenn etwa die letzte Version, die Sie compilierten `2.9-3` war, sollte Ihr rein binärer NMU die Versionsnummer `2.9-3+b1` tragen. Falls die letzte Version `3.4+b1` war (d.h. ein natives Paket mit einem vorhergehenden Neu-Compilierungs-NMU), sollte Ihr rein binärer NMU die Versionsnummer `3.4+b2`.[2] haben.

Ähnlich wie bei ersten Portierungs-Uploads ist der korrekte Weg **dpkg-buildpackage** aufzurufen `dpkg-buildpackage -B`, um nur die architekturabhängigen Teile des Pakets zu erstellen.

[2] In der Vergangenheit benutzten solche NMUs die dritte Stufe im Debian-Teil der Revisionsnummer, um ihren Status als reine Neu-Compilierung anzuzeigen. Diese Syntax war jedoch bei nativen Paketen mehrdeutig und erlaubte keine ordnungsgemäße Einordnung von reinen Neu-Compilierungs-NMUs, Quell-NMUs und Sicherheits-NMUs im gleichen Paket. Daher wurden sie verworfen und diese neue Syntax bevorzugt.

5.10.2.2 Wann ein Quell-NMU als Portierer gemacht werden sollte

Portierer, die einen Quell-NMU durchführen, folgen generell den Richtlinien, die unter Abschnitt 5.11 gefunden werden, genau wie nicht-Portierer. Es wird jedoch erwartet, dass der Wartezyklus für den Quell-NMU eines Portierers kleiner ist, als der von nicht Portierern, da Portierer mit einer großen Zahl von Paketen zurechtkommen müssen. Die Situation variiert wiederum abhängig von der Distribution, in die hochgeladen wird. Sie variiert außerdem in Abhängigkeit davon, ob die Architektur ein Kandidat für für die Einbindung in das nächste stabile Release ist. Die Release-Verwalter entscheiden welche Architekturen Kandidaten sind und kündigen dies an.

Falls Sie als Portierer einen NMU für `unstable` durchführen, sollten die vorher genannten Richtlinien der Portierung mit zwei Abwandlungen befolgt werden. Erstens ist die akzeptable Wartezeit – die Zeit zwischen dem Absenden des Fehlerberichts an das BTS und der Zeit, wenn es in Ordnung ist, einen NMU durchzuführen – sieben Tage für Portierer, die an der Distribution `unstable` arbeiten. Diese Zeitspanne kann verkürzt werden, falls das Problem kritisch ist und eine Notlage für die Portierungsanstrengung nach Ermessen der Gruppe der Portierer besteht. (Bedenken Sie, dass nichts davon in der Policy steht, sondern nur über Richtlinien vereinbart wurde.) Für Uploads nach `stable` oder `testing` stimmen Sie sich bitte zuerst mit dem Release-Team ab.

Zweitens sollten Protierer, die ein Quell-MMU durchführen, sicherstellen, dass der Fehler, den Sie an das BTS senden, den Schweregrad `serious` oder höher aufweist. Dies garantiert, dass ein einzelnes Quellpaket benutzt werden kann, um jede unterstützte Debian-Architektur zum Veröffentlichungszeitpunkt zu compilieren. Es ist sehr wichtig, dass es eine Version des Quell- und Binärpakets für alle Architekturen gibt, um vielen Lizenzen zu entsprechen.

Portierer sollten versuchen Patches zu vermeiden, die einfache Bastellösungen für Fehler in der aktuellen Version der Compiler-Umgebung, des Kernels oder der Libc bieten. Bisweilen sind solche Bastellösungen nicht hilfreich. Falls Sie an Compiler-Fehlern und dergleichen herumbasteln müssen, stellen Sie sicher, dass Sie Ihre Arbeit ordnungsgemäß in `#ifdef` einschließen. Dokumentieren Sie außerdem Ihren Murks, damit die Leute wissen, dass er entfernt werden muss, sobald die externen Probleme behoben wurden.

Portierer könnten außerdem einen inoffiziellen Ort haben, an dem sie die Ergebnisse Ihrer Arbeit während der Wartezeit ablegen. Dies hilft anderen, die aufder Portierung arbeiten, sogar während der Wartezeit aus der Arbeit des Portierers Nutzen zu ziehen. Natürlich haben solche Orte keinen offiziellen Segen oder Status, daher nehme sich der Käufer in Acht.

5.10.3 Portierungs-Infrastruktur und -Automatisierung

Es gibt eine Infrastruktur und mehrere Werkzeuge, die das Portieren von Paketen automatisieren. Dieser Abschnitt enthält einen kurzen Überblick dieser Automatisierung und Portierung mit diesen Werkzeugen. Lesen Sie die Paketdokumentation oder die Referenzen, um umfassende Informationen zu erhalten.

5.10.3.1 Mailinglisten und Web-Seiten

Web-Seiten, die den Status jeder Portierung enthalten, können unter `http://www.debian.org/ports/` gefunden werden.

Jede Portierung von Debian hat eine Mailingliste. Die Liste der Portierungs-Mailinglisten kann unter `http://lists.debian.org/ports.html` gefunden werden. Diese Listen werden benutzt, um die Arbeit der Portierer zu koordinieren und um eine Verbindung der Anwender der Portierung zu den Portierer herzustellen.

5.10.3.2 Werkzeuge der Portierers

Beschreibungen von vielen Werkzeugen für die Portierung können unter Abschnitt A.7 gefunden werden.

5.10.3.3 `wanna-build`

Das System `wanna-build` wird als ein verteiltes Client-/Server-Build-Verteilungssystem benutzt. Es wird üblicherweise zusammen mit Build-Daemons benutzt, die das Programm `buildd` ausführen. `Build daemons` sind »Slave«-Rechner, die das zentrale `wanna-build`-System kontaktieren, um eine Liste der Pakete zu beziehen, die erstellt werden müssen.

`wanna-build` ist noch nicht als Paket verfügbar. Alle Debian-Portierungsbestrebungen benutzen es jedoch zur automatisierten Paketerstellung. Das Werkzeug, das für die tatsächlichen Paket-Builds benutzt wird, `sbuild`, ist als Paket verfügbar. Lesen Sie dessen Beschreibung unter Abschnitt A.4.4. Bitte beachten Sie, dass die Paketversion nicht der des Build-Daemons entspricht, aber liegt ist nah genug bei dieser, um Probleme nachvollziehen zu können.

`wanna-build` ist generell für Portierer nützlich. Die meisten davon erzeugten Daten sind im Web unter `http://buildd.debian.org/` verfügbar. Diese Daten enthalten nächtlich aktualisierte Statistiken, Warteschlangeninformationen und Protokolle von Build-Versuchen.

Debian ist ziemlich stolz auf dieses System, da es so viele Verwendungsmöglichkeiten gibt. Unabhängige Entwicklergruppen können das System benutzen, um unterschiedliche Untergeschmacksrichtungen von Debian, die von allgemeinem Interesse sein können oder auch nicht (zum Beispiel eine Debian-Geschmacksrichtung, die mit »bounds checking« von **gcc** erstellt wurde) zu erstellen. Dadurch wird Debian auch in die Lage versetzt, ganze Distributionen schnell neu zu compilieren.

Das Wanna-Build-Team, das für Buildds verantwortlich ist, ist unter `debian-wb-team@lists.debian.org` erreichbar. Um festzustellen, wen (Wanna-Build-Team, Release-Team) Sie kontaktieren sollten und wie (E-Mail, BTS), sei auf `http://lists.debian.org/debian-project/2009/03/msg00096.html` verwiesen.

Wenn Sie um BinNMUs oder Give-Backs (erneute Versuche nach gescheitertem Build) ersuchen, benutzen Sie bitte das unter `http://release.debian.org/wanna-build.txt` beschriebene Format.

5.10.4 Wenn Ihr Paket *nicht* portierbar ist

Einige Pakete haben immer noch Probleme mit der Erstellung und/oder Ihrer Funktion auf einigen der von Debian unterstützten Architekturen und können überhaupt nicht oder nicht in einem akzeptablen Zeitraum portiert werden. Ein Beispiel ist ein Paket, das SVGA-spezifisch ist (nur auf `i386` und `amd64` verfügbar) oder andere Hardware-spezifische Funktionen benutzt, die nicht von allen Architekturen unterstützt werden.

Um zu verhindern, dass kaputte Pakete in das Archiv hochgeladen werden und Buildd-Zeit vergeuden, müssen Sie ein paar Dinge tun:

- Stellen Sie zuerst sicher, dass das Erstellen Ihres Pakets auf Architekturen, die es nicht unterstützt *fehlschlägt*. Es gibt mehrere Möglichkeiten dies zu bewirken. Der bevorzugte Weg ist es, während des Erstellens eine kleine Test-Suite zu verwenden, die die Funktionalität prüft und fehlschlägt, wenn es nicht funktioniert. Dies ist sowieso eine gute Idee, da es (einige) kaputte Uploads auf allen Architekturen verhindert und außerdem ermöglicht, das Paket zu erstellen, sobald die benötigte Funktionalität verfügbar ist.

  Zusätzlich sollten Sie in `debian/control` `any` auf eine Liste der unterstützten Architekturen ändern, falls Sie glauben, die Liste der unterstützten Architekturen sei ziemlich gleichbleibend. Auf diese Art wird das Erstellen ebenfalls fehlschlagen und dies einem menschlichen Leser ohne tatsächliche Versuche anzeigen.

- Um zu verhindern, dass Autobuilder unnötig versuchen Ihr Paket zu erstellen, muss es in `Packages-arch-specific` enthalten sein, einer Liste, die vom **wanna-build**-Skript benutzt wird. Die aktuelle Version ist unter `http://buildd.debian.org/quinn-diff/Packages-arch-specific` verfügbar. Bitte lesen am Anfang der Datei, wer wegen der Änderungen kontaktiert wird.

Bitte beachten Sie, dass es nicht ausreicht, Ihr Paket nur zu `Packages-arch-specific` hinzuzufügen ohne dafür zu sorgen, dass das Erstellen auf nicht unterstützten Architekturen fehlschlägt: Ein Portierer oder jemand anderes, der versucht Ihr Paket zu erstellen, könnte Ihr Paket fälschlicherweise hochladen ohne zu bemerken, dass es nicht funktioniert. Wenn in der Vergangenheit einige Binärpakete auf nicht unterstützte Architekturen hochgeladen wurden, bitten Sie um Ihre Entfernung, indem Sie einen Fehlerbericht gegen `ftp.debian.org` einreichen.

5.10.5 Unfreie Pakete als automatisch erstellbar kennzeichnen

Standardmäßig werden Pakete aus dem Bereich `non-free` nicht durch das Autobuilder-Netzwerk gebaut (meistens, weil die Lizenz der Pakete dem entgegen stehen könnte). Um zu aktivieren, dass ein Paket gebaut wird, müssen Sie die folgenden Schritte durchführen:

1. Prüfen, ob es rechtlich erlaubt und technisch möglich ist, das Paket automatisch zu bauen;
2. `XS-Autobuild:yes` zu den Kopfzeilen von `debian/control` hinzufügen;
3. eine E-Mail an nonfree@release.debian.org senden und erklären, warum das Paket rechtlich und technisch automatisch gebaut werden kann.

5.11 Non-Maintainer Uploads (NMUs)

Jedes Paket hat einen oder mehrere Betreuer. Normalerweise sind das Leute, die daran arbeiten und neue Versionen des Pakets hochladen. In einigen Situationen ist es nützlich, dass auch andere Entwickler neue Versionen hochladen können, zum Beispiel, falls sie einen Fehler in einem Paket beheben möchten, das sie nicht betreuen, wenn der Betreuer Hilfe benötigt, um auf Probleme zu antworten. Solche Uploads werden *Non-Maintainer Uploads (NMU)* genannt.

5.11.1 Wann und wie ein NMU durchgeführt wird

Beachten Sie die folgenden Fragen, bevor Sie einen NMU durchführen:

- Behebt Ihr NMU wirklich Fehler? Kosmetische Probleme zu beheben oder den Paketierungsstil in NMUs zu ändern ist unerwünscht.
- Haben Sie dem Paketbetreuer genug Zeit gegeben? Wann wurde der Fehler an das BTS gemeldet? Es ist nicht unüblich, für eine oder zwei Wochen beschäftigt zu sein. Ist der Fehler so schwer, dass er jetzt sofort behoben werden muss oder kann er noch ein paar Tage warten?
- Wie überzeugt sind Sie von Ihren Änderungen? Bitte erinnern Sie sich an den hippokratischen Eid: »Schaden Sie vor allem nicht«. Es ist besser, ein Paket mit einem offenen schweren Fehler zu belassen, als einen nicht funktionierenden Patch darauf anzuwenden oder einen, der den Fehler versteckt, anstatt Ihn zu beheben. Falls Sie nicht 100% sicher sind, was Sie getan haben, könnte es eine gute Idee sein, den Rat anderer zu suchen. Vergessen Sie nicht, das viele Leute sauer sind, falls Ihr NMU etwas kaputt macht.
- Haben Sie Ihre Absicht, einen NMU durchzuführen, zumindest im BTS klar ausgedrückt? Es ist außerdem ratsam zu versuchen, den Paketbetreuer auf andere Arten zu kontaktieren (private E-Mail, IRC).
- Haben Sie versucht den Betreuer zu kontaktieren, falls er normalerweise aktiv und zugänglich ist? Im Allgemeinen sollte es als wünschenswert erachtet werden, dass sich ein Betreuer selbst um ein Problem kümmert und dass er die Möglichkeit hat, Ihr Patch zu überprüfen und zu korrigieren, da er potentielle Probleme kennen sollte, die demjenigen fehlen könnten, der den NMU durchführt. Die Zeit wird meist besser investiert, wenn dem Betreuer die Gelegenheit gegeben wird, eine Fehlerbehebung selbst hochzuladen.

Wenn Sie einen NMU durchführen, sollten Sie zuerst dafür sorgen, dass Ihre Absicht einen NMU durchzuführen klar ist. Dann müssen Sie einen Patch mit den Unterschieden zwischen dem aktuellen Paket und dem geplanten NMU an das BTS senden. Das Skript **nmudiff** im Paket `devscripts` könnte hilfreich sein.

Während Sie den Patch vorbereiten, sollten Sie besser einige paketspezifischen Verfahren kennen, die der Betreuer möglicherweise benutzt. Ihn einzubeziehen verringert die Belastung, die Änderungen zurück in den normalen Arbeitsablauf des Pakets zu integrieren und vergrößert daher die Möglichkeit, dass das geschieht. Ein guter Ort, um etwas über die paketspezifischen Methoden zu erfahren, ist `debian/README.source`.

Sofern Sie keinen ausgezeichneten Grund haben, dies nicht zu tun, müssen Sie dem Paketbetreuer Zeit zum Reagieren geben (zum Beispiel durch Hochladen in die `DELAYED`-Warteschlange). Hier sind einige empfohlene Werte, die für Verzögerungen benutzt werden:

- Der Upload behebt nur veröffentlichungskritische Fehler, die älter als sieben Tage sind, ohne Betreueraktivität beim Fehler für sieben Tage und ohne Hinweis, dass eine Fehlerbehebung im Gang ist: 0 Tage
- Upload, der nur veröffentlichungskritische Fehler behebt, die älter als sieben Tage sind: zwei Tage
- Upload, der nur veröffentlichungskritische Fehler und Fehler mit Schweregrad »important« behebt: fünf Tage
- Andere NMUs: zehn Tage

Diese Verzögerungen sind nur Beispiele. In manchen Fällen, wie bei Uploads, die Sicherheitsprobleme beheben oder der Behebung belangloser Fehler, die einen Übergang blockieren, ist es wünschenswert, dass ein repariertes Paket `unstable` eher erreicht.

Manchmal entscheiden Release-Verwalter NMUs mit kürzeren Verzögerungen für eine Untermenge von Fehlern zu erlauben (z.B. veröffentlichungskritische Fehler, die älter als sieben Tage sind). Außerdem führen manche Paketbetreuer sie selbst in der Liste LowThresholdNmu (niedrige Schwelle für NMUs) auf und akzeptieren, dass NMUs ohne Verzögerung hochgeladen werden. Aber sogar in diesen Fällen ist es immer noch ratsam, dem Betreuer ein paar Tage Zeit zum Reagieren zu geben bevor Sie etwas hochladen, insbesondere, wenn der Patch vorher nicht im BTS verfügbar war oder falls Sie wissen, dass der Paketgetreuer allgemein aktiv ist.

Nachdem Sie einen NMU hochgeladen haben, sind Sie für mögliche Probleme verantwortlich, die Sie möglicherweise eingeleitet haben. Sie müssen das Paket im Auge behalten (ein gute Möglichkeit dies zu erreichen, ist es, das Paket im PTS zu abonnieren).

Dies ist keine Lizenz, rücksichtslos NMUs durchzuführen. Falls Sie einen NMU auf den Weg bringen, wenn es klar ist, dass die Betreuer aktiv sind und ein Patch zeitnah anerkennen würden oder falls Sie die Empfehlungen dieses Dokuments ignorieren, könnte Ihr Upload zu einem Konflikt mit dem Betreuer führen. Sie sollten immer darauf vorbereitet sein, im Nachhinein für den von Ihnen durchgeführten NMU aus eigener Kraft einstehen zu können.

5.11.2 NMUs und `debian/changelog`

Genauso wie jeder andere (Quell-) Upload, müssen NMUs einen Eintrag in `debian/changelog` hinzufügen, der mitteilt, was mit diesem Upload geändert wurde. Die erste Zeile muss explizit erwähnen, dass dieser Upload ein NMU ist, z.B:

```
  * Non-maintainer upload.
```

Die Möglichkeiten der Versionsvergabe für NMUs unterscheidet sich bei nativen und nicht nativen Paketen.

Falls es sich um ein natives Paket (ohne Debian-Revision in der Versionsnummer) handelt, muss die Versionsnummer die des letzten Betreuer-Uploads plus `+nmu`*X* sein, wobei *X* ein Zähler ist, der bei `1` beginnt. Falls der letzte Upload auch ein NMU war, wird der Zähler erhöht. Wenn beispielsweise die aktuelle Version `1.5` ist, dann würde der NMU die Version `1.5+nmu1` werden.

Falls es sich um kein natives Paket handelt, sollten Sie eine untergeordnete Versionsnummer zum Debian-Revisionsteil der Versionsnummer hinzufügen (der Teil nach dem letzten Bindestrich). Diese zusätzliche Zahl muss bei `1` anfangen. Wenn zum Beispiel die aktuelle Version `1.5-2` ist, dann würde ein NMU die Version `1.5-2.1` erhalten. Falls eine neue Originalversion im NMU paketiert wird, wird die Debian-Revision auf `0` gesetzt, zum Beispiel `1.6-0.1`.

In beiden Fällen sollte der Zähler erhöht werden, falls der letzte Upload auch ein NMU war. Wenn zum Beispiel die letzte Version `1.5+nmu3` war (ein natives Paket, für das bereits ein NMU durchgeführt wurde), würde der NMU die Version `1.5+nmu4` erhalten.

Es wird ein spezielles Schema der Versionsvergabe benötigt, um zu verhindern dass die Arbeit des Maintainers unterbrochen wird, da die Benutzung einer Ganzzahl für die Debian-Revision einen potentiellen Konflikt mit einem Betreuer-Upload hervorruft, der bereits zur Zeit des NMUs vorbereitet wird oder sogar in der Ftp-Warteschlange NEW ist. Es hat außerdem den Vorteil, dass es optisch klar zeigt, dass ein Paket im Archiv nicht vom offiziellen Betreuer stammt.

Falls Sie ein Paket nach Testing oder Stable hochladen, müssen Sie manchmal den Baum der Versionsnummern verzweigen. Dies ist beispielsweise bei Sicherheitsaktualisierungen der Fall. Aus diesem Grund sollte eine Version der Form `+deb`*XY*`u`*Z* benutzt werden, wobei *X* und *Y* die Major- und Minor-Release-Nummern und *Z* ein Zähler ist, der bei `1` beginnt. Wenn die Release-Nummer noch nicht bekannt ist (öfters der Fall bei `testing` zu Beginn des Release-Zyklus), muss die kleinste Release-Nummer benutzt werden, die größer als die letzte Stable-Release-Nummer ist. Während zum Beispiel Lenny (Debian 5.0) Stable ist, hätte ein Sicherheits-NMU für Stable für ein Paket der Version `1.5-3` die Version `1.5-3+deb50u1`, während ein Sicherheits-NMU für Squeeze die Version `1.5-3+deb60u1` bekäme. Nach dem Release von Squeeze werden Sicherheits-Uploads in die Distribution `testing` mit der Versionsnummer `+deb61uZ` versehen, bis bekannt ist, ob das Release Debian 6.1 oder Debian 7.0 sein wird (falls dies der Fall ist, bekommen Uploads die Version `+deb70uZ`).

5.11.3 Benutzung der Warteschlange `DELAYED/`

Nachdem Sie um Erlaubnis ersucht haben, einen NMU durchzuführen, ist es ineffizient, auf eine Antwort zu warten, da es für denjenigen, der den NMU durchführt, einen Kontextwechsel zurück zu diesem Thema erfordert. Die Warteschlange `DELAYED` (siehe Abschnitt 5.6.2) ermöglicht es einem Entwickler einen NMU und alle nötigen Aufgaben gleichzeitig durchzuführen. Sie sollten zum Beispiel anstatt dem Betreuer mitzuteilen, dass Sie das aktualisierte Paket in sieben Tagen hochladen werden, das Paket nach `DELAYED/7` hochladen und dem Betreuer mitteilen, dass er sieben Tage zum reagieren hat. Während dieser Zeit kann der Betreuer Sie bitten, den Upload etwas länger aufzuschieben oder Ihren Upload abzubrechen.

Die Warteschlange `DELAYED` sollte nicht benutzt werden, um zusätzlichen Druck auf den Paketbetreuer auszuüben. Es ist besonders wichtig, dass Sie erreichbar sind, um die Verzögerung des Uploads abzubrechen, bevor die Zeit abläuft, da der Betreuer den Upload nicht selbst abbrechen kann.

Falls Sie einen NMU nach `DELAYED` durchführen und der Betreuer sein Paket vor Ablauf der Verzögerung aktualisiert, wird Ihr Upload abgelehnt, da bereits eine neuere Version im Archiv verfügbar ist. Idealerweise achtet der Betreuer darauf, dass er die von Ihnen vorgeschlagenen Änderungen (oder zumindest eine Lösung für die Probleme, die sie behandeln) in diesen Upload einfließen lässt.

5.11.4 NMUs aus Sicht des Paketbetreuers

Wenn jemand einen NMU Ihres Pakets dürchführt, bedeutet dies, dass er Ihnen helfen möchte, es in einem guten Zustand zu halten. Dies beschert den Anwendern schneller reparierte Pakete. Sie könnten überlegen, ob Sie denjenigen, der das NMU durchführte, fragen möchten, ob er Mitbetreuer des Pakets werden will. Der Erhalt eines NMUs für ein Paket ist keine schlechte Sache; es bedeutet nur, dass das Paket interessant genug ist, dass andere Leute daran arbeiten.

Um einen NMU anzuerkennen, schließen Sie dessen Änderungen und Änderungsprotokolleinträge in Ihren nächsten Upload ein. Falls Sie den NMU nicht anerkennen, schließen Sie den Änderungsprotokolleintrag des NMUs in Ihr Änderungsprotokoll ein, die Fehler bleiben im BTS geschlossen, werden aber als Ihre Betreuerversion des Pakets betreffend aufgeführt.

5.11.5 Quell-NMUs gegenüber rein binären NMUs (binNMUs)

Der vollständige Name eines NMUs ist *Quell-NMU*. Es gibt auch einen anderen Typ, der *rein binärer NMU* oder *binNMU* genannt wird. Ein binNMU ist ebenfalls ein Paket-Upload durch jemand anderes als den Paketbetreuer. Er ist jedoch rein binär.

Wenn eine Bibliothek (oder andere Abhängigkeit) aktualisiert wird, könnte es notwendig sein, das Paket neu zu erstellen. Da keine Änderungen am Quellcode nötig sind, wird das gleiche Quellpaket benutzt.

BinNMUs werden üblicherweise auf Buildds durch Wanna-Build ausgelöst. `debian/changelog` wird ein Eintrag hinzugeführt, der erklärt, warum der Upload nötig war und die Versionsnummer wird, wie in Abschnitt 5.10.2.1 beschrieben, erhöht. Dieser Eintrag sollte nicht im nächsten Upload enthalten sein.

Buildds lädt Pakete für seine Architektur als rein binäre Uploads in das Archiv. Genaugenommen sind dies binNMUs. Sie werden jedoch normalerweise nicht als NMU bezeichnet und fügen `debian/changelog` keinen Eintrag hinzu.

5.11.6 NMUs gegenüber QS-Uploads

NMUs sind Uploads von Paketen durch jemand anderes als den ihnen zugewiesenen Betreuer. Es gibt noch einen anderen Upload-Typ, bei dem ihm das hochgeladene Paket nicht gehört: QS-Uploads. QS-Uploads sind Uploads verwaister Pakete.

QS-Uploads sind normalen Betreuer-Uploads sehr ähnlich: sie können alles reparieren, sogar kleine Probleme. Die Versionsnummerierung ist normal und es sind keine verzögerten Uploads nötig. Der Unterschied besteht darin, dass Sie nicht als `Maintainer` oder `Uploader` des Pakets aufgeführt werden. Außerdem hat der Änderungsprotokolleintrag eines QS-Uploads eine spezielle erste Zeile:

```
* QA upload.
```

Falls Sie einen NMU durchführen möchten und es so aussieht, als sei der Betreuer nicht aktiv, ist es vernünftig zu prüfen, ob das Paket verwaist ist (diese Information wird auf der Seite des Pakets im Paketverfolgungssystem »PTS« angezeigt). Beim ersten QS-Upload zu einem verwaisten Paket, sollte der Betreuer auf `Debian QA Group <packages@qa.debian.org>` gesetzt werden. Bei verwaisten Paketen, für die noch kein QS-Upload durchgeführt wurde, ist immer noch der alte Betreuer gesetzt. Eine Liste dieser Pakete finden Sie unter http://qa.debian.org/orphaned.html.

Anstatt einen QS-Upload durchzuführen, können Sie auch erwägen das Paket zu adoptieren, indem Sie sich selbst zum Betreuer machen. Sie benötigen von niemandem eine Erlaubnis, um ein verwaistes Paket zu adoptieren, Sie müssen sich nur selbst als Betreuer einsetzen und die neue Version hochladen (siehe Abschnitt 5.9.5).

5.11.7 NMUs gegenüber Team-Uploads

Manchmal reparieren und/oder aktualisieren Sie ein Paket, weil Sie Mitglied des Paketierungs-Teams sind (das als `Maintainer` oder `Uploader` eine Mailingliste benutzt, siehe Abschnitt 5.12), aber Sie möchten sich selbst nicht zu den `Uploaders` hinzufügen, da Sie nicht planen, regulär an diesem speziellen Paket mitzuarbeiten. Falls dies mit den Richtlinien Ihres Teams in Einklang steht, können Sie einen normalen Upload durchführen ohne direkt als

Maintainer oder Uploader aufgeführt zu werden. In diesem Fall sollten Sie Ihren Änderungsprotokolleintrag mit der folgenden Zeile beginnen:

```
* Team upload.
```

5.12 Gemeinschaftliche Verwaltung

Gemeinschaftliche Verwaltung ist ein Begriff, der die gemeinsamen Verwaltungspflichten von Debian-Paketen durch mehrere Leute beschreibt. Diese Zusammenarbeit ist fast immer eine gute Idee, da sie generell in einer höheren Qualität und einer schnelleren Fehlerbehebungszeit resultiert. Es wird dringend empfohlen, dass Pakete, die die Priorität standard haben, oder Teil der grundlegenden Zusammenstellung sind, Mitbetreuer haben.

Generell gibt es einen Hauptbetreuer und einen oder mehrere Mitbetreuer. Der Hauptbetreuer ist die Person, deren Name im Feld Maintainer der Datei debian/control steht. Mitbetreuer sind alle anderen Betreuer. Sie werden normalerweise in der Datei debian/control im Feld Uploaders aufgeführt.

In seiner grundlegendsten Form ist der Prozess neue Mitbetreuer hinzuzufügen ziemlich einfach:

- Richten Sie den Mitbetreuer mit Zugriff auf die Quellen ein, aus denen Sie das Paket erstellen. Dies impliziert, dass Sie ein netzwerkfähiges Versionskontrollsystem wie CVS oder Subversion benutzen. Unter anderem stellt Alioth solche Werkzeuge bereit (siehe Abschnitt 4.12).
- Fügen Sie im Feld Uploaders im ersten Absatz der Datei debian/control den korrekten Namen und die Adresse des Mitbetreuers ein.

  ```
  Uploaders: John Buzz <jbuzz@debian.org>, Adam Rex <arex@debian.org>
  ```

- Wird das PTS (Abschnitt 4.10) benutzt, sollten sich die Mitbetreuer selbst für das entsprechende Quellpaket einschreiben.

Eine andere Form der gemeinschaftlichen Verwaltung stellt die Team-Verwaltung dar. Sie wird empfohlen, falls Sie mehrere Pakete mit der gleichen Entwicklergruppe verwalten. In diesem Fall müssen die Felder Maintainer und Uploaders jedes Pakets mit Vorsicht verwaltet werden. Es wird empfohlen eines der beiden folgenden Schemen auszuwählen:

1. Setzen Sie den Hauptverantwortlichen für das Paket in das Feld Maintainer ein. In Uploaders werden die Adresse der Mailingliste und die Team-Mitglieder, die sich um das Paket kümmern, eingetragen.
2. Setzen Sie die Adresse der Mailingliste in das Feld Maintainer ein. Im Feld Uploaders werden die Team-Mitglieder eingetragen, die sich um das Paket kümmern. In diesem Fall müssen Sie sicherstellen, dass die Mailingliste Fehlerberichte ohne menschliches Eingreifen akzeptiert (wie Moderation für Nicht-Abonnenten).

Es ist jedenfalls ein schlechter Einfall, automatisch alle Team-Mitglieder in das Feld Uploaders einzutragen. Es überfüllt die Paketübersicht des Entwicklers (siehe Abschnitt 4.11) mit Paketen, um die sich nicht wirklich jemand kümmert und erweckt den falschen Eindruck einer guten Betreuung. Aus dem gleichen Grund müssen sich Team-Mitglieder nicht selbst im Feld Uploaders hinzufügen, nur weil sie das Paket einmal hochgeladen haben. Sie können einen Team-Upload durchführen (siehe Abschnitt 5.11.7). Im umgekehrten Fall ist es eine schlechte Idee ein Paket mit nur der Adresse der Mailingliste im Feld Maintainer zu behalten und keinen Uploaders.

5.13 Die Distribution Testing

5.13.1 Grundlagen

Pakete werden normalerweise in die Distribution Testing installiert, nachdem sie einem gewissen Maß von testing in unstable unterzogen wurden.

Sie müssen auf allen Architekturen synchron sein sein und dürfen keine Abhängigkeiten haben, die sie uninstallierbar machen; sie dürfen außerdem zum Zeitpunkt, an dem sie in testing installiert werden, keine bekannten veröffentlichungskritischen Fehler haben. Auf diese Art sollte testing immer ein potentieller Release-Kandidat sein. Bitte lesen Sie das Folgende, um weitere Einzelheiten zu erfahren.

5.13.2 Aktualisierungen von Unstable

Die Skripte, die die Distribution `testing` aktualisieren, werden zweimal täglich ausgeführt, gleich nach der Installation der aktualisierten Pakete. Diese Skripte werden `britney` genannt. Sie generieren die `Packages`-Dateien für die Distribution `testing`, aber sie tun dies auf eine clevere Art; sie versuchen jede Unstimmigkeit zu vermeiden und nur fehlerfreie Pakete zu benutzen.

Die Aufnahme eines Pakets von `unstable` ist durch Folgendes bedingt:

- Das Paket muss zwei, fünf oder zehn Tage in `unstable` verfügbar gewesen sein, abhängig von der Dringlichkeit (hoch, mittel oder niedrig). Bitte beachten Sie, dass die Dringlichkeit unnachgiebig ist, was bedeutet, dass die höchste Dringlichkeit mit der seit dem letzten Übergang nach `testing` hochgeladen wurde, berücksichtigt wird. Diese Verzögerungen können während eines Freeze verdoppelt werden oder alle Übergänge nach `testing` könnten zusammen deaktiviert werden.
- Es darf keine veröffentlichungskritischen Fehler haben (veröffentlichungskritische Fehler betreffend die in `unstable` verfügbare Version, aber nicht die in `testing`);
- Es muss auf allen Architekturen verfügbar sein, auf denen es vorher in `unstable` erstellt wurde. Um diese Information zu prüfen, könnte dak ls von Interesse sein.
- Es darf keine Abhängigkeiten von Paketen zerstören, die bereits in `testing` verfügbar sind.
- Die Pakete, von denen es abhängt, müssen entweder in `testing` verfügbar sein oder sie müssen zur gleichen Zeit in `testing` akzeptiert werden (und das werden sie, falls sie alle nötigen Kriterien erfüllen).

Um herauszufinden, ob ein Paket nach `testing` fortschreitet oder nicht, sehen Sie die Ausgabe des `testing`-Skripts auf der Web-Seite Debian »Testing«-Distribution oder benutzen Sie das Programm **grep-excuses** aus dem Paket `devscripts`. Dieses Hilfswerkzeug kann einfach in einer crontab(5) benutzt werden, um sich selbst auf dem aktuellen Stand über den Fortschritt des Pakets nach `testing` zu informieren.

Die Datei `update_excuses` gibt nicht immer den genauen Grund an, weshalb ein Paket abgelehnt wurde. Sie können es selbst herausfinden, indem Sie schauen, was durch die Aufnahme des Pakets zerstört würde. Die Web-Seite Debian »Testing«-Distribution stellt einige weitere Informationen über die üblichen Probleme bereit, die derartigen Ärger verursachen.

Manchmal errreichen einige Pakete `testing` niemals, da die Zusammensetzung wechselseitiger Beziehungen zu kompliziert ist und durch das Skript nicht aufgelöst werden können. Im Folgenden finden Sie Einzelheiten.

Einige weitere Abhängigkeitsanalysen werden unter `http://release.debian.org/migration/` angezeigt — aber seien Sie gewarnt, diese Seite zeigt außerdem Build-Abhängigkeiten, die nicht von Britney berücksichtigt werden.

5.13.2.1 Veraltet

Für das `testing`-Migrations-Skript bedeutet veraltet: Es gibt in `unstable` verschiedene Versionen für die Release-Architektur (außer für Architekturen in Fuckedarches; Fuckedarches ist eine Liste von Architekturen, die nicht weitergeführt werden (in `update_out.py`), aber gegenwärtig ist sie leer). Veraltet hat nichts damit zu tun, welche Architekturen dieses Paket in `testing` unterstützt.

Sehen Sie sich dieses Beispiel an:

	alpha	**arm**
testing	1	-
unstable	1	2

Das Paket ist auf `alpha` in `unstable` veraltet und wird nicht nach `testing` gelangen. Das Paket zu entfernen würde überhaupt nicht helfen. Das Paket ist immer noch auf `alpha` veraltet und wird sich nicht nach `testing` ausbreiten.

Falls FTP-Master jedoch ein Paket in `unstable` entfernt (hier auf `arm`):

	alpha	**arm**	**hurd-i386**
testing	1	1	-
unstable	2	-	1

In diesem Fall ist das Paket auf allen Architekturen in `unstable` aktuell (und das zusätzliche `hurd-i386`

tut nichts zur Sache, da es keine Release-Architektur ist).

Manchmal kommt die Frage auf, ob es möglich ist, Pakete aufzunehmen, die noch nicht auf allen Architekturen erstellt wurden: Nein. Einfach nur nein. (Außer Sie betreuen Glibc oder so).

5.13.2.2 Entfernen aus Testing

Manchmal wird ein Paket entfernt, um einem anderen Paket die Aufnahme zu gewähren: Dies geschieht nur, um einem *anderen* Paket die Aufnahme zu gewähren, falls es in jedem anderen Sinn in Ordnung ist. Angenommen, `a` könnte z.B. nicht mit der neuen Version von `b` installiert werden, dann könnte `a` entfernt werden, um `b` die Aufnahme zu ermöglichen.

Natürlich gibt es einen anderen Grund, ein Paket aus `testing` zu entfernen: Es ist einfach zu fehlerhaft (und ein einfacher veröffentlichungskritischer Fehler reicht nicht aus, um diesen Status zu bekommen).

Wenn ein Paket außerdem aus `unstable` entfernt wurde und kein Paket in `testing` mehr davon abhängt, dann wird es automatisch entfernt.

5.13.2.3 Wechselseitige Abhängigkeiten

Eine Situation, die nicht sehr gut von Britney gehandhabt wird, ist, wenn Paket `a` von einer neuen Version des Pakets `b` abhängt und umgekehrt.

Ein Beispiel hierfür:

	testing	**unstable**
a	1; depends: b=1	2; depends: b=2
b	1; depends: a=1	2; depends: a=2

Weder Paket `a` noch Paket `b` wird für die Aktualisierung berücksichtigt.

Aktuell erfordert dies einige manuelle Eingriffe des Release-Teams. Bitte kontaktieren Sie es per E-Mail an debian-release@lists.debian.org, falls dies bei einem Ihrer Pakete auftritt.

5.13.2.4 Beeinflussen eines Pakets in Testing

Generell gibt es keine Bedeutung des Status eines Pakets in `testing`, der das Hinüberwechseln der nächsten Version eines Pakets von `unstable` nach `testing` erfordert, mit zwei Ausnahmen: Falls ein Paket veröffentlichungsunkritischer wird, könnte es sogar aufgenommen werden, wenn es immer noch veröffentlichungskritisch ist. Die zweite Ausnahme ist, wenn die Version des Pakets in `testing` auf den verschiedenene Architekturen nicht mehr synchron ist: Dann könnte für jede Architektur nur ein Upgrade auf die Version des Quellpakets durchgeführt werden; dies kann jedoch nur auftreten, wenn das Paket vorher dorthin durchgedrängt wurde, die Architektur in Fuckedarches ist oder kein binäres Paket dieser Architektur in `unstable` bei der Migration nach `testing` vorhanden war.

Zusammengefasst heißt das: Der einzige Einfluss eines Pakets, das sich in `testing` befindet, auf eine neue Version des gleichen Pakets besteht darin, dass die neue Version leichter aufgenommen werden kann.

5.13.2.5 Eintelheiten

Falls Sie die Einzelheiten interessieren, erklärt dies, wie Britney funktioniert:

Die Pakete werden betrachtet, um festzulegen, ob Sie gültige Kandidaten sind. Dies ergibt die Aktualisierungs-Ausreden. Die häufigsten Gründe, warum ein Paket nicht berücksichtigt wird lauten: zu neu, zu viele veröffentlichungskritische Fehler und auf einigen Architekturen veraltet. Für diesen Teil von Britney verfügen die Release-Verwalter über Druckmittel verschiedener Stärke, um eine Berücksichtigung des Pakets durch Britney zu erzwingen. (Außerdem ist der grundlegende Freeze in diesen Teil von Britney einprogrammiert.) (Es gibt ein ähnliches Ding für rein binäre Aktualisierungen, aber dies wird hier nicht beschrieben. Falls Sie dies interessiert, sehen Sie bitte den Code durch.)

Nun kommt der komplexere Teil: Britney versucht `testing` mit den gültigen Kandidaten zu aktualisieren. Dazu versucht Britney, jeden gültigen Kandidaten zur Distribution `testing` hinzuzufügen. Falls die Zahl nicht installierbarer Pakete in `testing` sich nicht erhöht, wird das Paket akzeptiert. Ab diesem Zeitpunkt wird das Paket als Teil von `testing` betrachtet, so dass alle anschließenden Installierbarkeitstests dieses Paket einbeziehen. Hinweise des Release-Teams werden, abhängig vom genauen Typ, vor diesem Hauptdurchlauf verarbeitet.

Falls Sie weitere Einzelheiten suchen, können Sie unter `http://ftp-master.debian.org/testing/update_output/` nachsehen.

Die Hinweise sind unter `http://ftp-master.debian.org/testing/hints/` verfügbar.

5.13.3 Direkte Aktualisierungen für Testing

Die Distribution `testing` wird, den genannten Regeln folgend, mit Paketen aus `unstable` gespeist. In einigen Fällen ist es jedoch nötig, Pakete hochzuladen, die nur für `testing` erstellt wurden. Dafür empfiehlt es sich, nach `testing-proposed-updates` hochzuladen.

Merken Sie sich, dass dorthin hochgeladene Pakete nicht automatisch verarbeitet werden, sie müssen erst durch die Hand des Release-Verwalters gehen. Daher sollten sie besser über einen triftigen Grund verfügen, dorthin hochzuladen. Um zu erfahren, was in den Augen der Release-Verwalter ein triftiger Grund ist, sollten sie die Anweisungen lesen, die sie reglemäßig auf debian-devel-announce@lists.debian.org erteilen.

Sie sollten nicht nach `testing-proposed-updates` hochladen, wenn Sie Ihre Pakete über `unstable` aktualisieren können. Falls Sie dies nicht können (zum Beispiel, weil Sie eine neuere Entwicklerversion in `unstable` haben), könnten Sie diese Einrichtung nutzen, aber es ist empfohlen, dass Sie zuerst die Release-Verwalter um Erlaubnis fragen. Aktualisierungen über `unstable` sind sogar möglich, wenn ein Paket eingefroren ist, falls der Upload über `unstable` keine neuen Abhängigkeiten mit sich bringt.

Versionsnummern werden normalerweise durch Hinzufügen einer fortlaufenden Nummer zum Codenamen der `testing`-Distribution gebildet, wie `1.2squeeze1` für den ersten Upload über `testing-proposed-updates` der Paketversion `1.2`.

Bitte stellen Sie sicher, dass keines dieser Elemente in Ihrem Upload fehlt:

- Vergewissern Sie sich, dass Ihr Paket wirklich `testing-proposed-updates` durchlaufen muss und nicht über `unstable` gehen kann.
- Achten Sie darauf, dass Sie nur die kleinstmögliche Anzahl von Änderungen eingefügt haben.
- Sorgen Sie dafür, dass das Änderungsprotokoll eine entsprechende Erklärung enthält.
- Überzeugen Sie sich, dass in Ihre Zieldistribution `testing` oder `testing-proposed-updates` geschrieben haben.
- Prüfen Sie nach, ob Sie Ihr Paket in `testing` und nicht in `unstable` getestet haben.
- Gehen sie sicher, dass Ihre Versionsnummer höher als die Version in `testing` und `testing-proposed-updates` ist und niedriger als in `unstable`.
- Nach dem Hochladen und erfolgreichen Erstellen auf allen Plattformen, kontaktieren Sie das Team unter debian-release@lists.debian.org und ersuchen Sie es um Genehmigung Ihres Uploads.

5.13.4 Häufig gestellte Fragen

5.13.4.1 Was sind veröffentlichungskritische Fehler und wie werden Sie gezählt?

Alle Fehler mit einem höheren Schweregrad werden standardmäßig als veröffentlichungskritisch angesehen. Aktuell sind dies Fehler der Schweregrade `critical`, `grave` und `serious`.

Von solchen Fehlern wird angenommen, dass sie einen Einfluss darauf haben, ob das Paket mit dem `stable`-Release von Debian veröffentlicht wird: Im Allgemeinen würde ein Paket, das offene veröffentlichungskritische Fehler hat, nicht nach `testing` gelangen und demzufolge nicht in `stable` veröffentlicht werden.

Als `unstable`-Fehleranzahl werden alle veröffentlichungskritischen Fehler gezählt, die als *`Paket`*-/*`Versions`*-Kombinationen zugehörig markiert sind, die in Unstable für eine Release-Architektur verfügbar sind. Die Fehleranzahl in `testing` ist sinngemäß definiert.

5.13.4.2 Wie kann das Installieren eines Pakets in `testing` andere Pakete möglicherweise zerstören?

Die Struktur der Distributionsarchive ist so aufgebaut, dass Sie nur eine Version eines Pakets enthalten kann. Ein Paket wird durch seinen Namen definiert. Wenn also das Quellpaket `acmefoo` zusammen mit seinen Binärpaketen `acme-foo-bin`, `acme-bar-bin`, `libacme-foo1` und `libacme-foo-dev` nach `testing` installiert wird, wird die alte Version entfernt.

Die alte Version könnte jedoch ein Binärpaket mit einem alten Soname einer Bibliothek bereitstellen, wie `libacme-foo0`. Das Entfernen der alten `acmefoo` wird `libacme-foo0` entfernen, was alle Pakete zerstört, die davon abhängen.

Offenbar betrifft dies hauptsächlich Pakete, die ändernde Zusammenstellungen von Binärpaketen in unterschiedlichen Versionen bereitstellen (wiederum hauptsächlich Bibliotheken). Es wird jedoch außerdem Pakete betreffen, deren Versionsabhängigkeiten über die Varianten ==, <= oder << deklariert wurden.

Wenn sich die Zusammenstellung von Binärpaketen, die von einem Quellpaket bereitgestellt werden, auf diese Weise ändert, müssen alle Pakete, die von den alten Programmen abhängen, aktualisiert werden, damit sie stattdessen von den neuen Programmen abhängen. Da das Installieren eines solchen Quellpakets in `testing` alle Pakete zerstört, die in `testing` davon abhängen, ist nun auf Folgendes zu achten: All die abhängigen Pakete müssen aktualisiert werden und selbst bereit zur Installation sein, so dass sie nicht kaputt gehen. Sobald alles bereit ist, ist normalerweise ein manuelles Eingreifen des Release-Verwalters oder eines Assistenten nötig.

Falls Sie Probleme mit komplizierten Gruppen von Paketen wie diesem haben, kontaktieren Sie debian-devel@lists.debian.org oder debian-release@lists.debian.org, um Hilfe zu erhalten.

Kapitel 6

Optimale Vorgehensweise beim Paketieren

Debians Qualität ist größtenteils den Debian-Richtlinien zu verdanken, die explizit grundlegende Anforderungen definieren, die alle Debian-Pakete erfüllen müssen. Bisher gibt es außerdem eine gemeinsame Geschichte der Erfahrung, die hinter den Debian-Richtlinien steckt, einer Ansammlung jahrelanger Erfahrung im Paketieren. Viele sehr talentierte Leute haben großartige Werkzeuge geschaffen, Werkzeuge, die Ihnen als Debian-Betreuer helfen, ausgezeichnete Pakete zu erstellen und zu pflegen.

Dieses Kapitel stellt einige optimale Vorgehensweisen für Debian-Entwickler vor. Das alles sind lediglich Empfehlungen und keine Anforderungen oder Richtlinien. Dies sind nur einige subjektive Hinweise, Ratschläge und Fingerzeige, die von Debian-Entwicklern gesammelt wurden. Suchen Sie sich einfach das heraus, was Ihnen am meisten zusagt.

6.1 Optimale Vorgehensweisen für `debian/rules`

Die folgenden Empfehlungen gelten für die Datei `debian/rules`. Da `debian/rules` den Build-Prozess steuert und die Dateien auswählt, die in das Paket gelangen (direkt oder indirekt), ist es normerweise die Datei, der die Betreuer die meiste Zeit widmen.

6.1.1 Helfer-Skripte

Der Grund für die Benutzung von Helfer-Skripten in `debian/rules` ist, dass sie den Betreuern eine geteilte gemeinsame Logik inmitten vieler Pakete einräumen. Nehmen Sie zum Beispiel die Frage, wie Menü-Einträge installiert werden: Sie müssen die Datei in `/usr/share/menu` (oder `/usr/lib/menu` für ausführbare binäre Menü-Dateien, wenn nötig) ablegen und den Betreuerskripten Befehle hinzufügen, um Menü-Einträge zu registrieren bzw. ihre Registrierung zu entfernen. Dies ist eine sehr häufige Tätigkeit für Pakete. Warum sollte daher jeder Betreuer all dies für sich selbst neu schreiben und dabei möglicherweise Fehler verursachen? Außerdem, den Fall gesetzt, das Menü-Verzeichnis würde sich ändern, dann müsste jedes Paket geändert werden.

Helfer-Skripte kümmern sich um diese Probleme. Angenommen, Sie erfüllen alle Gepflogenheiten, die das Helfer-Skript erwartet, dann kümmert sich das Helfer-Skript um alle Einzelheiten. Änderungen an den Richtlinien können im Helfer-Skript erledigt werden. Dann müssen Pakete nur mit der neuen Version des Helfer-Skripts erstellt und sonst nicht geändert werden.

Anhang A enthält ein paar verschiedene Helfer-Skripte. Das gängigste und beste (nach Meinung von Debian) Helfersystem ist `debhelper`. Verhergehende Helfersysteme, wie `debmake` waren monolithisch. Sie konnten nicht den Teil des Helfers herausgreifen und auswählen, den Sie nützlich fanden, mussten aber den Helfer für alles benutzen. `debhelper` besteht jedoch aus mehreren getrennten kleinen **dh_***-Programmen. **dh_installman** installiert und komprimiert zum Beispiel Handbuchseiten, **dh_installmenu** installiert Menü-Dateien und so weiter. Daher bietet es eine ausreichende Flexibilität, die kleinen Helfer-Skripte dort zu benutzen, wo sie nützlich sind in Verbindung mit handgemachten Befehlen in `debian/rules`.

Sie können mit `debhelper` anfangen, indem Sie debhelper(1) lesen und sich die Beispiele ansehen, die dem Paket beigefügt sind. **dh_make** aus dem Paket `dh-make` (siehe Abschnitt A.3.2) kann benutzt werden, um ein einfaches Quellpaket in ein mit `debhelper` bearbeitetes Paket umzuwandeln. Gleichwohl sollte diese Kurzform Sie jedoch nicht davon überzeugen, dass Sie sich nicht plagen müssen, um die einzelnen **dh_***-Helfer zu verstehen. Falls Sie einen Helfer benutzen möchten, müssen Sie sich die Zeit nehmen zu lernen, wie dieser Helfer benutzt wird, um zu verstehen was er erwartet und wie er sich verhält.

Einige Personen sind der Ansicht, dass einfache `debian/rules`-Dateien besser sind, da Sie nicht die Feinheiten eines Helfer-Systems erlernen müssen. Diese Entscheidung liegt allein bei Ihnen. Benutzen Sie das, was für Sie am besten klappt. Unter http://arch.debian.org/arch/private/srivasta/ sind viele Beispiele für einfache `debian/rules`-Dateien verfügbar.

6.1.2 Unterteilen Sie Ihre Patches in mehrere Dateien

Große, komplexe Pakete könnten mehrere Fehler haben, die Sie bewältigen müssen. Falls Sie mehrere Fehler direkt in der Quelle beheben und nicht sorgfältig vorgehen, kann es schwierig werden, die verschiedenen Patche, die Sie bereitgestellt haben, zu unterscheiden. Es kann ziemlich chaotisch werden, wenn Sie das Paket auf eine neue Originalversion aktualisieren müssen, die einige (aber nicht alle) Reparaturen enthält. Sie können nicht die komplette Zusammenstellung der Diffs nehmen (z.B. aus `.diff.gz`) und austüfteln, welcher Patch es als Einheit zurücksetzt, da Fehler im Original repariert wurden.

Glücklicherweise ist es nun mit dem Quellformat »3.0 (quilt)« möglich, die Patche getrennt zu halten ohne `debian/rules` zur Einrichtung eines Patch-Systems ändern zu müssen. Patche werden in `debian/patches/` gespeichert und wenn das Quellpaket entpackt wird, werden automatisch die Patches angewandt, die in `debian/patches/series` aufgeführt sind. Wie der Name schon sagt, können Patches mit **quilt** verwaltet werden.

Wenn Sie die ältere Quelle »1.0« verwenden, ist es auch möglich, Patche zu trennen, aber es muss ein zugehöriges Patch-System verwandt werden: Die Patch-Dateien werden innerhalb der Debian-Patch-Datei (`.diff.gz`) mitgeliefert, normalerweise im Verzeichnis `debian/`. Der einzige Unterschied ist, dass sie nicht unmittelbar von **dpkg-source** angewandt werden, sondern von der `build`-Regel der Datei `debian/rules` durch eine Abhängigkeit in der `patch`-Regel. Im Gegenzug werden sie von der Regel `clean` durch eine Abhängigkeit zur Regel `unpatch` umgekehrt.

quilt ist das dafür empfohlene Werkzeug. Es erledigt alles oben beschriebene und ermöglicht außerdem die Verwaltung von Patch-Serien. Weitere Informationen finden Sie im Paket `quilt`.

Es gibt noch andere Werkzeuge, um Patches zu verwalten, wie **dpatch** und das in `cdbs` eingebaute Patch-System.

6.1.3 Pakete mit mehreren Binärdateien

Ein einzelnes Quellpaket wird oft mehrere Binärpakete erstellen, entweder um mehrere Geschmacksrichtungen der gleichen Software bereitzustellen (z.B. das `vim`-Quellpaket) oder um mehrere kleine Pakete anstelle eines einzelnen großen zu erzeugen (z,B. damit der Benutzer nur die benötigte Untermenge installieren kann und dadurch Plattenplatz spart).

Der zweite Fall kann einfach in `debian/rules` verwaltet werden. Sie müssen nur die entsprechenden Dateien aus dem Build-Verzeichnis in die entsprechenden temporären Baumstrukturen des Pakets verschieben. Dies können Sie mit **install** oder **dh_install** aus `debhelper` erledigen. Sorgen Sie dafür, dass Sie die unterschiedlichen Umsetzungen der verschiedenen Pakete prüfen, um sicherzustellen, dass Sie die wechselseitigen Abhängigkeiten in `debian/control` richtig gesetzt haben.

Der erste Fall ist etwas schwieriger, da er mehrmaliges Neukompilieren der selben Software einschließt, aber mit unterschiedlichen Konfigurationsoptionen. Das `vim`-Quellpaket ist ein Beispiel dafür, wie dies mit einer handgemachten `debian/rules`-Datei verwaltet wird.

6.2 Optimale Vorgehensweisen für `debian/control`

Die folgenden Vorgehensweisen sind maßgeblich für die Datei `debian/control`. Sie ergänzen die Richtlinien für Paketbeschreibungen.

Die Beschreibung des Pakets, wie sie durch das entsprechende Feld in der Datei `control` definiert wird, enthält sowohl die Paketübersicht, als auch die ausführliche Beschreibung des Pakets. Abschnitt 6.2.1 beschreibt übliche Richtlinien für beide Teile der Paketbeschreibung. Diesen folgend stellt Abschnitt 6.2.2 Richtlinien speziell für die Übersicht bereit und Abschnitt 6.2.3 enthält spezielle Richtlinien für die Beschreibung.

6.2.1 Allgemeine Richtlinien für Paketbeschreibungen

Die Paketbeschreibung sollte für den erwarteten Durchschnittsanwender geschrieben werden, der Durchschnittsperson, die das Paket benutzt und davon profitiert. Entwicklungspakete sind beispielsweise für Entwickler und können in einer technischen Sprache verfasst werden. Anwendungen für allgemeinere Zwecke, wie Editoren, sollten für Anwender mit weniger technischem Verständnis geschrieben werden.

Die Durchsicht der Paketbeschreibungen mündet in der Schlussfolgerung, dass die meisten Paketbeschreibungen technischer Natur sind, also nicht so geschrieben, das sie für Anwender ohne technischen Hintergrund einen Sinn ergeben. Sofern Ihr Paket nicht wirklich für technikkundige Anwender ist, ist dies ein Problem.

Wie können sie für nicht technikkundige Anwender schreiben? Vermeiden Sie Fachsprache. Vermeiden Sie, sich auf Anwendungen und Rahmenwerke zu beziehen, mit denen der Anwender möglicherweise nicht vertraut ist – GNOME oder KDE sind in Ordnung, da Anwender wahrscheinlich mit diesen Begriffen vertraut sind, aber GTK+ vermutlich nicht. Versuchen Sie nicht irgendein Wissen vorauszusetzen, geben Sie eine Einführung.

Seien Sie objektiv. Paketbeschreibungen sind nicht der richtige Ort, um Ihr Paket zu verfechten, egal wie Sie sehr sie es mögen. Denken Sie daran, dass der Leser nicht die gleichen Dinge wichtig nimmt, wie Sie.

Bezüge zu Namen anderer Softwarepakete, Protokollnamen, Standards oder Spezifikationen sollten, falls sie existiert, in ihrer vorschriftsmäßigen Form verwandt werden. Benutzen Sie zum Beispiel X Window System, X11 oder X, nicht X Windows, X-Windows, or X Window. Benutzen Sie GTK+, nicht GTK oder gtk. Benutzen Sie GNOME nicht Gnome. Benutzen Sie PostScript, nicht Postscript oder postscript.

Falls Sie Probleme beim Verfassen Ihrer Beschreibung haben, könnten Sie sie an debian-l10n-english@lists.debian.org senden und um Rückmeldung ersuchen.

6.2.2 Die Paketübersicht oder Kurzbeschreibung

Die Richtlinie sagt, die Übersichtszeile (die Kurzbeschreibung) muss kurz sein, darf den Paketnamen nicht wiederholen, muss aber auch informativ sein.

Die Übersichtsfunktionen ist ein Ausdruck, der das Paket beschreibt, kein kompletter Satz, daher ist Zeichensetzung unangebracht: Es wird weder eine besondere Großschreibung noch ein abschließender Punkt benötigt. Außerdem sollten jegliche bestimmten und unbestimmten Artikel am Anfang weggelassen werden – »a«, »an« oder »the«. Deshalb zum Beispiel:

```
Package: libeg0
Description: exemplification support library
```

Technisch gesehen ist dies eine Nominalphrase im Gegensatz zu einer Verbalphrase. Eine gute Entscheidungsregel ist, dass es möglich sein sollte, den Paket-*Namen* und die *Übersicht* in diesem Schema zu ersetzen:

The package *Name* provides {a,an,the,some} *Übersicht*.

Zusammenstellungen verwandter Pakete können ein alternatives Schema benutzen, das die Übersicht in zwei Teile unterteilt, als erstes eine Beschreibung der ganzen Suite und als zweites eine Zusammenfassung der Rolle, die das Paket darin spielt:

```
Package: eg-tools
Description: simple exemplification system (utilities)

Package: eg-doc
Description: simple exemplification system - documentation
```

Diese Übersicht folgt einem geänderten Schema. Wo ein Paket »*Name*« die Übersicht »*Suite* (*Rolle*)« oder »*Suite* - *Rolle*« hat, sollten die Elemente so ausgedrückt werden, dass sie in dieses Schema passen:

The package *name* provides {a,an,the} *role* for the *suite*.

6.2.3 Die ausführliche Beschreibung

Die ausführliche Beschreibung ist die wichtigste für den Benutzer verfügbare Information über ein Paket, bevor er es installiert. Sie sollte alle nötigen Informationen bereitstellen, damit der Anwender entscheiden kann, ob er das Paket installiert. Es ist anzunehmen, dass der Benutzer bereits die Paketübersicht gelesen hat.

Die ausführliche Beschreibung sollte aus vollständigen Sätzen bestehen.

Der erste Absatz der ausführlichen Beschreibung sollte die folgenden Fragen beantworten: Was tut das Paket? Bei welchen Aufgaben hilft es dem Anwender? Es ist wichtig dies auf eine nicht technische Weise zu beschreiben, sogar dann, wenn die Zielgruppe des Pakets notwendigerweise einen technischen Hintergrund hat.

Die folgenden Absätze sollten die folgenden Fragen beantworten: Warum benötige ich als Benutzer dieses Paket? Welche anderen Funktionen bietet das Paket? Welche herausragenden Funktionen und Mängel gibt es im Vergleich zu anderen Paketen (z.B. falls Sie X benötigen, benutzen Sie Y)? Steht das Paket in einem Zusammenhang mit anderen Paketen, die nicht vom Paketmanager gehandhabt werden (z.B. ist dies der Client für den Foo-Server)?

Seien Sie vorsichtig, um Rechtschreib- und Grammatikfehler zu vermeiden. Sorgen Sie für eine Rechtschreibprüfung. Sowohl **ispell** als auch **aspell** haben spezielle Modi zur Prüfung von `debian/control`-Dateien:

```
ispell -d american -g debian/control
```

```
aspell -d en -D -c debian/control
```

Anwender erwarten normalerweise, dass diese Fragen in der Paketbeschreibung beantwortet werden:

- Was tut das Paket? Falls es eine Erweiterung eines anderen Pakets ist, dann sollte eines Kurzbeschreibung des Pakets, das es erweitert, hier eingefügt werden.
- Warum sollte ich dieses Paket wollen? Dies bezieht sich auf das vorhergehende, aber nicht das gleiche (dies ist ein Mail-Client. Er ist toll, schnell, hat Schnittstellen zu PGP, LDAP und IMAP, hat die Funktionen X, Y und Z).
- Falls dieses Paket nicht direkt installiert werden sollte, sondern von einem anderen Paket mitinstalliert wird, sollte die erwähnt werden.
- Falls das Paket `experimental` ist oder es andere Gründe gibt, weshalb es nicht benutzt werden sollte und wenn es andere Pakete gibt, die stattdessen benutzt werden sollen, sollte dies auch hier stehen.
- Was unterscheidet dieses Paket von der Konkurrenz? Ist es eine bessere Implementierung? Mehr Funktionen? Andere Funktionen? Warum sollte die Wahl auf dieses Paket fallen?

6.2.4 Homepage der Originalautoren

Es wird empfohlen, dass Sie die URL der Homepage des Pakets in das Feld `Homepage` im Abschnitt `Source` von `debian/control` hinzufügen. Diese Information in die Paketbeschreibung selbst einzutragen, wird als missbilligt angesehen.

6.2.5 Ort des Versionsverwaltungssystems

Es gibt zusätzliche Felder für den Ort des Versionsverwaltungssystems in `debian/control`.

6.2.5.1 Vcs-Browser

Der Wert dieses Feldes sollte eine `http://`-URL sein, die auf eine via Web durchstöberbare Kopie des Versionsverwaltungssystems-Depots verweist, das benutzt wird, um das angegebene Paket zu verwalten, falls verfügbar.

Die Information ist dazu gedacht, dem Endanwender zu nutzen, der gewillt ist, die letzte am Paket geleistete Arbeit zu durchstöbern (z.B. wenn nach einem Patch gesucht wird, der einen Fehler behebt, der in der Fehlerdatenbank als `pending` gekennzeichnet ist).

6.2.5.2 Vcs-*

Der Wert dieses Feldes sollte eine Zeichenkette sein, die den Ort des Versionsverwaltungssystem-Depots eindeutig identifiziert, das zur Verwaltung des angegebenen Pakets benutzt wird, falls verfügbar. `*` identifiziert das Versionsverwaltungssystem. Aktuell werden die folgenden vom Paketverfolgungssystem unterstützt: `arch`, `bzr` (Bazaar), `cvs`, `darcs`, `git`, `hg` (Mercurial), `mtn` (Monotone) und `svn` (Subversion). Es ist erlaubt, mehrere unterschiedliche Versionsverwaltungssystem-Felder für das gleiche Paket anzugeben: Sie werden alle auf der PTS-Web-Schnittstelle angezeigt.

Die Information ist für Benutzer bestimmt, die im gegebenen Versionsverwaltungssystem sachkundig sind und bereit, die aktuelle Version des Pakets aus den VCS-Quellen zu erstellen. Andere Nutzungen dieser Information könnten das automatische Erstellen der letzen VCS-Version des Pakets umfassen. Dazu sollte der vom Feld angegebene Ort, die Version nicht kennen und auf den Hauptzweig zeigen (bei Versionsverwaltungssystemen, die dieses Konzept unterstützen). Außerdem sollte der Endanwender auf den Ort, auf den verwiesen wird, zugreifen können. Die Erfüllung dieser Anforderungen könnte einen anonymen Zugriff voraussetzen, statt auf eine Version davon zu verweisen, auf die über SSH zugegriffen wird.

Im folgenden Beispiel wird ein Fall von einem Feld eines Subversions-Depots des Pakets `vim` gezeigt. Beachten Sie, wie die URL im `svn://`-Schema aufgebaut ist (im Gegensatz zu `svn+ssh://`) und wie sie auf den Zweig `trunk/` zeigt. Außerdem wird die Benutzung der Felder `Vcs-Browser` und `Homepage`, wie oben beschrieben, ebenfalls gezeigt.

```
Source: vim
Section: editors
Priority: optional
<snip>
Vcs-Svn: svn://svn.debian.org/svn/pkg-vim/trunk/packages/vim
Vcs-Browser: http://svn.debian.org/wsvn/pkg-vim/trunk/packages/vim
Homepage: http://www.vim.org
```

6.3 Optimale Vorgehensweisen für `debian/changelog`

Die folgenden Vorgehensweisen ergänzen die Richtlinien für Änderungsprotokolldateien.

6.3.1 Verfassen nützlicher Änderungsprotokolleinträge

Der Änderungsprotokolleintrag einer Paketüberarbeitung dokumentiert Änderungen in nur dieser Überarbeitung. Der Schwerpunkt liegt auf der Beschreibung bedeutender und für den Anwender sichtbarer Änderungen, die seit der letzten Version vorgenommen wurden.

Der Fokus liegt darauf, *was* geändert wurde – wer, wie und wann ist normalerweise nicht so wichtig. Erinnern Sie gleichwohl höflich an die Leute, die merklich Hilfe beim Erstellen des Pakets geleistet haben (die z.B. Patche gesandt haben).

Es ist nicht nötig, die belanglosen und offensichtlichen Änderungen näher auszuführen. Sie können außerdem mehrere Änderungen in einem Eintrag zusammenfassen. Fassen Sie sich andererseits nicht zu kurz, falls Sie eine größere Änderung vorgenommen haben. Stellen Sie insbesondere klar, falls es Änderungen gibt, die das Verhalten des Programms ändern. Benutzen Sie für weitere Erklärungen die Datei `README.Debian`.

Benutzen Sie geläufiges Englisch, so dass die Mehrheit der Leser es begreifen kann. Vermeiden Sie Abkürzungen, technische Begriffe und Fachsprache, wenn Sie Änderungen erklären, die Fehlerberichte schließen, insbesondere bei Fehlern, die von Anwendern eingereicht wurden, die Ihnen als technisch unerfahren aufgefallen sind. Seien Sie höflich, fluchen Sie nicht.

Manchmal ist es wünschenswert, den Änderungsprotokolleinträgen die Namen der Dateien voranzustellen, die geändert wurden. Es ist jedoch nicht nötig, explizit jede einzelne geänderte Datei aufzuführen, insbesondere dann nicht, wenn die Änderung klein oder wiederholend war. Sie können Platzhalter verwenden.

Treffen Sie keine Annahmen, wenn Sie sich auf Fehler beziehen. Sagen Sie, welches Problem vorlag, wie es behoben wurde und hängen Sie die Zeichenkette »closes: #nnnnn« an. Weitere Informatioenn erhalten Sie unter Abschnitt 5.8.4.

6.3.2 Häufige Missverständnisse über Änderungsprotokolleinträge

Die Änderungsprotokolleinträge sollten **keine** allgemeinen Paketierungsthemen dokumentieren (Hey, falls Sie die Foo.conf suchen, die ist in /etc/blah/.), da von Administratoren und Anwendern angenommen wird, dass sie zumindest entfernt damit vertraut sind, wie solche Dinge im Allgemeinen auf Debian-Systemen eingerichtet sind. Erwähnen Sie jedoch, wenn Sie den Ort einer Konfigurationsdatei ändern.

Die einzigen Fehler, die mit einem Änderungsprotokolleintrag geschlossen werden, sollten Fehler sein, die tatsächlich in der gleichen Überarbeitung des Pakets behoben werden. Das Schließen von Fehlern ohne Bezug dazu, ist eine falsche Vorgehensweise. Siehe Abschnitt 5.8.4.

Die Änderungsprotokolleinträge sollten **nicht** für zufällige Diskussionen mit Leuten, die Fehler melden (Ich kann keine Schutzverletzungen sehen, wenn ich Foo mit der Option Bar starte. Senden Sie weitere Informationen.), allgemeine Äußerungen über das Leben, das Universum und alles mögliche (Entschuldigung, dass der Upload so lange brauchte, aber ich hatte die Grippe.) oder Hilfeersuchen (Die Fehlerliste für dieses Paket ist riesig, bitte packen Sie mit an) benutzt werden. Solche Dinge werden normalerweise nicht von Ihrer Zielgruppe bemerkt, könnten aber viele Leute stören, die Informationen über tatsächliche Änderungen am Paket lesen möchten. Weitere Informationen über die Benutzung der Fehlerdatenbank finden Sie unter Abschnitt 5.8.2.

Es ist ein alter Brauch im ersten regulären Upload des Paketbetreuers das Beheben von Fehlern durch Non-Maintainer-Uploads zu bestätigen. Da Debian nun über eine Versionsverwaltung verfügt, reicht es aus, die NMU-Änderungsprotokolleinträge aufzubewahren und diese Tatsache nur in Ihrem eigenen Änderungsprotokolleintrag zu erwähnen.

6.3.3 Häufige Fehler in Änderungsprotokolleinträgen

Die folgenden Beispiele demonstrieren einige häufige Fehler oder Beispiele für schlechten Stil in Änderungsprotokolleinträgen.

```
  * Fixed all outstanding bugs.
```

Dies teilt den Lesern offensichtlich nichts Nützliches mit.

```
  * Applied patch from Jane Random.
```

Was war das für ein Patch?

```
  * Late night install target overhaul.
```

Welche Reparatur wurde ausgeführt? Soll die Erwähnung der späten Nacht daran erinnern, dass man dem Code nicht trauen sollte?

```
  * Fix vsync FU w/ ancient CRTs.
```

Zu viele Abkürzungen und es ist nicht klar, wovon der äh ... Mist (Hoppla, ein Schimpfwort) tatsächlich handelt oder wie er repariert wurde.

```
  * This is not a bug, closes: #nnnnnn.
```

Erst einmal ist es absolut unnötig, das Paket hochzuladen, um diese Information zu übermitteln. Benutzen Sie stattdessen die Fehlerdatenbank. Zweitens fehlt die Erklärung, warum dieser Bericht kein Fehler ist.

```
  * Has been fixed for ages, but I forgot to close; closes: #54321.
```

Falls Sie aus irgend einem Grund die Fehlernummer in einem früheren Änderungsprotokolleintrag nicht erwähnt haben, ist das kein Problem. Schließen Sie den Fehler einfach im BTS. Es ist nicht nötig, die Änderungsprotokolldatei anzurühren, vorausgesetzt, die Beschreibung der Fehlerbehebung ist bereits darin enthalten (Dies gilt auch für Reparaturen durch die Originalautoren/-Betreuer. Sie müssen keine Fehler verfolgen, die diese bereits vor Jahren in Ihrem Änderungsprotokoll behoben haben).

```
  * Closes: #12345, #12346, #15432
```

Wo ist die Beschreibung? Falls Ihnen keine aussagekräftige Nachricht einfällt, beginnen Sie, die Titel der verschiedenen Fehler einzufügen.

6.3.4 Änderungsprotokolle mit `NEWS.Debian`-Dateien ergänzen

Wichtige Nachrichten über Änderungen in einem Paket können auch in die `NEWS.Debian`-Dateien geschrieben werden. Die Nachrichten werden durch Werkzeuge wie `apt-listchanges` vor dem ganzen Rest des Änderungsprotokolls angezeigt. Dies ist das vorzugsweise Mittel, dem Anwender bedeutende Änderungen in einem Paket mitzuteilen. Es ist besser, als `debconf`-Notizen zu benutzen, da es weniger stört und der Anwender nach der Installation zurückgehen und in der `NEWS.Debian`-Datei nachschlagen kann. Es ist auch besser, als die Hauptänderungen in `README.Debian` aufzuführen, da der Anwender solche Notizen leicht übersehen kann.

Das Dateiformat entspricht dem des Änderungsprotokolls, die Sternchen werden allerdings weggelassen und jedes Nachrichtenelement wird, wenn nötig, mit einem vollständigen Satz beschrieben, statt der kurz gefassten Zusammenfassungen, die in ein Änderungsprotokoll einfließen. Sie sind gut beraten, Ihre Datei durch `dpkg-parsechangelog` laufen zu lassen, um ihre Formatierung zu prüfen, da sie nicht automatisch während des Builds getestet wird, so wie dies beim Änderungsprotokoll getan wird. Hier nun ein Beispiel einer echten `NEWS.Debian`-Datei:

```
cron (3.0pl1-74) unstable; urgency=low

    The checksecurity script is no longer included with the cron package:
    it now has its own package, checksecurity. If you liked the
    functionality provided with that script, please install the new
    package.

 -- Steve Greenland <stevegr@debian.org>  Sat,  6 Sep 2003 17:15:03 -0500
```

Die Datei `NEWS.Debian` wird als `/usr/share/doc/Paket/NEWS.Debian.gz` installiert. Sie ist komprimiert und hat immer diesen Namen, auch in nativen Debian-Paketen. Falls Sie `debhelper` benutzen, wird `dh_installchangelogs` die `debian/NEWS`-Dateien für Sie installieren.

Anders als Änderungsprotokolldateien, müssen Sie die `debian/NEWS`-Dateien nicht bei jeder Veröffentlichung aktualisieren. Aktualisieren Sie sie nur, wenn Sie etwas besonders berichtenswertes haben, worüber der Anwender Bescheid wissen sollte. Falls Sie überhaupt keine Nachrichten haben, ist es nicht nötig, Ihrem Paket eine `debian/NEWS`-Datei mitzugeben. Keine Nachricht ist eine gute Nachricht!

6.4 Optimale Vorgehensweisen für Betreuerskripte

Betreuerskripte beinhalten die Dateien `debian/postinst`, `debian/preinst`, `debian/prerm` und `debian/postrm`. Diese Skripte geben auf jede Installations- oder Deinstallationseinrichtung des Pakets acht, die bloß durch Erstellen und Entfernen von Dateien und Verzeichnissen gehandhabt wird. Die folgenden Anweisungen ergänzen die Debian Policy.

Betreuerskripte müssen Idempotent sein. Dies bedeuted, dass Sie sicherstellen müssen, dass nichts Schlimmes passiert, wenn das Skript zweimal aufgerufen wird, während es normalerweise einmal aufgerufen würde.

Standardein- und -ausgabe könnten zu Protokollierungszwecken umgeleitet werden (z.B. in Pipes), verlassen Sie sich daher nicht darauf, dass sie ein Terminal sind.

Jegliche Bedienerführung oder interaktive Konfiguration sollte so gering wie möglich gehalten werden. Wenn es nötig ist, sollten Sie das Paket `debconf` für die Schnittstelle benutzen. Denken Sie daran, dass diese Bedienerführung nur in der `configure`-Stufe des `postinst`-Skripts stattfinden kann.

Halten Sie die Betreuerskripte so einfach wie möglich. Es wird empfohlen, nur reine POSIX-Shell-Skripte zu benutzen. Falls Sie irgendwelche Bash-Funktionen benötigen, vergessen Sie nicht, dass das Betreuerskript eine Shebang-Zeile haben muss. POSIX-Shell oder Bash werden für Perl bevorzugt, da sie `debhelper` ermöglichen, den Skripten einfach Teile hinzuzufügen.

Falls Sie Ihre Betreuerskripte ändern, stellen Sie sicher, dass Sie das Entfernen des Pakets, die mehrmalige Installation und das vollständige Entfernen testen. Vergewissern Sie sich, dass nach dem vollständigen Entfernen des Pakets alles komplett weg ist, sprich, es muss jede erzeugte Datei entfernen, die direkt oder indirekt in irgendeinem Betreuerskript erstellt wurde.

Falls Sie prüfen möchten, ob ein Befehl existiert, sollten Sie so etwas benutzen:

```
if [ -x /usr/sbin/install-docs ]; then ...
```

Falls Sie nicht dem Pfad eines Befehls fest ins Betreuerskript schreiben möchten, könnte die folgende POSIX-konforme Shell-Funktion helfen:

```
pathfind() {
    OLDIFS="$IFS"
    IFS=:
    for p in $PATH; do
        if [ -x "$p/$*" ]; then
            IFS="$OLDIFS"
            return 0
        fi
    done
    IFS="$OLDIFS"
    return 1
}
```

Sie können diese Funktion benutzen, um `$PATH` nach einem Befehlsnamen zu durchsuchen, der als Argument übergegeben wird. Sie gibt »true« (null) zurück, falls der Befehl gefunden wurde und »false«, falls nicht. Dies ist wirklich die portierbarste Möglichkeit, da `command -v`, **type** und **which** nicht POSIX-konform sind.

Obwohl **which** eine akzeptable Alternative ist, da es aus dem benötigten Paket `debianutils` stammt, liegt es nicht auf der Wurzel-Partition. Zumindest liegt es eher in `/usr/bin` als in `/bin`, so dass es nicht in Skripten benutzt werden kann, die vor dem Einhängen von `/usr` ausgeführt werden. Dieses Problem werden allerdings die meisten Skripte nicht haben.

6.5 Konfigurationsverwaltung mit `debconf`

`Debconf` ist ein Konfigurationsverwaltungssystem, das von allerlei Paketierungsskripten (hauptsächlich `postinst`) benutzt werden kann, um Rückmeldungen von Anwendern betreffend der Paketkonfiguration abzufragen. Direkte Benutzer-Interaktion muss nun zugunsten der Interaktion mit `debconf` vermieden werden. Dies wird in Zukunft nicht interaktive Installationen ermöglichen.

Debconf ist ein großartiges Werkzeug, aber es wird oft mangelhaft benutzt. Viele alltägliche Fehler sind auf der Handbuchseite debconf-devel(7) aufgeführt. Manchmal ist es nötig zu lesen, falls Sie sich entscheiden Debconf zu benutzen. Außerdem werden hier ein paar optimale Vorgehensweisen vorgestellt.

Diese Richtlinien enthalten einige Schreibstil- und Typografie-Empfehlungen, allgemeine Betrachtungen über die Benutzung von Debconf, ebenso wie spezifischere Empfehlungen für einige Teile der Distribution (das Installationssystem besispielsweise).

6.5.1 Missbrauchen Sie Debconf nicht

Seit Debconf in Debian erschien, wurde es verbreitet missbraucht und viel von der Kritik, die bei der Debian-Distribution einging, rührte vom Debconf-Missbrauch mit der Notwendigkeit, ein großes Fragenbündel zu beantworten, bevor eine Kleinigkeit installiert war.

Behalten Sie Aufrufnotizen dort, wozu sie gehören: den Dateien `NEWS.Debian` oder `README.Debian`. Benutzen Sie nur Notizen für wichtige Anmerkungen, die direkt die Benutzbarkeit des Pakets beeinflussen. Bedenken Sie, dass Notizen die Installation immer blockieren, bis Sie bestätigt wurden oder den Anwender per E-Mail bleästigt haben.

Wählen Sie die Prioritäten der Fragen in Betreuerskripten sorgfältig. Einzelheiten über Prioritäten finden Sie unter debconf-devel(7). Die meisten Fragen sollten mittlere oder niedrige Prioritäten nutzen.

6.5.2 Allgemeine Empfehlungen für Autoren und Übersetzer

6.5.2.1 Schreiben Sie korrektes Englisch.

Die meisten Debian-Paketbetreuer haben nicht Englisch als Muttersprache. Daher ist es für sie nicht einfach, korrekt formulierte Schablonen zu verfassen.

Bitte benutzen (und missbrauchen) Sie die Mailingliste debian-l10n-english@lists.debian.org. Lassen Sie Ihre Schablonen korrekturlesen.

Schlecht geschriebene Schablonen werfen ein armseliges Bild auf Ihr Paket, Ihre Arbeit ... oder sogar auf Debian selbst.

Vermeiden Sie soweit möglich technische Fachsprache. Wenn sich einige Begriffe für Sie vertraut anhören, könnten sie für andere unverständlich sein. Falls sie sich nicht vermeiden lassen, versuchen Sie sie zu erklären (benutzen Sie die längere Beschreibung). Versuchen Sie dabei, zwischen Aussagekraft und Einfachheit abzuwägen.

6.5.2.2 Seien sie nett zu Übersetzern

Debconf-Schablonen können übersetzt werden. Debconf bietet zusammen mit seinem Schwesterpaket **po-debconf** ein einfaches Gerüst, um Schablonen durch Übersetzer-Teams oder sogar einzelne Personen übersetzen zu lassen.

Bitte benutzen Sie Gettext-basierte Schablonen. Installieren Sie `po-debconf` auf Ihrem Entwicklungssystem und lesen Sie dessen Dokumentation (**man po-debconf** ist ein guter Anfang).

Vermeiden Sie, Schablonen häufig zu ändern. Das Ändern von Schablonen führt zu Mehrarbeit für Übersetzer, deren Übersetzungen unvollständig werden. Eine unvollständige Übersetzung ist eine Zeichenkette, bei der sich seit dem Übersetzen das Original geändert hat und das daher eine Aktualisierung durch den Übersetzer benötigt. Wenn die Änderungen klein genug sind, wird die Originalübersetzung in den PO-Dateien beibehalten und mit `fuzzy` gekennzeichnet.

Falls Sie planen, Änderungen an Ihren Originalschablonen vorzunehmen, benutzen Sie bitte das Benachrichtigungssystem namens **podebconf-report-po**, das vom Paket `po-debconf` bereitgestellt wird, um die Übersetzer zu kontaktieren. Die meisten aktiven Übersetzer sind sehr zugänglich und Ihre Arbeit zusammen mit Ihren geänderten Schablonen einzubeziehen, wird Sie vor zusätzlichen Uploads bewahren. Falls Sie Gettext-basierte Schablonen verwenden, werden die Namen und E-Mail-Adressen der Übersetzer in den Kopfzeilen der PO-Dateien erwähnt und von **podebconf-report-po** benutzt.

Ein empfohlene Art, das Hilfswerkzeug zu benutzen ist:

```
cd debian/po && podebconf-report-po --call --languageteam --withtranslators -- ↩
    deadline="+10 days"
```

Dieser Befehl wird zuerst die PO- und POT-Dateien in `debian/po` mit dem Schablonendateien synchronisieren, die in `debian/po/POTFILES.in` aufgeführt sind. Dann wird er einen Aufruf für neue Übersetzungen an die Mailingliste debian-i18n@lists.debian.org senden. Am Schluss wird er außerdem einen Aufruf für neue Übersetzungen an das Sprach-Team (im Feld `Language-Team` jeder PO-Datei erwähnt) sowie den letzten Übersetzer (erwähnt in `Last-translator`) senden.

Es wird immer gewürdigt, wenn Sie den Übersetzern einen Abgabetermin geben, so dass sie ihre Arbeit organisieren können. Bitte denken Sie daran, dass einige Übersetzer-Teams einen formalisierten Übersetzung-/Korrekturprozess haben und eine Zeitspanne, die kürzer als zehn Tage ist, als unangemessen angesehen wird. Eine kürzere Frist übt zuviel Druck auf die Übersetzer-Teams aus und sollte für sehr kleine Änderungen genommen werden.

Im Zweifelsfall können Sie auch das Übersetzer-Team für eine bestimmte Sprache (debian-l10n-xxxxx@lists.debian.org) oder die Mailingliste debian-i18n@lists.debian.org kontaktieren.

6.5.2.3 Entfernen Sie die Fuzzy-Markierungen in vollständigen Übersetzungen, wenn Sie Tipp- und Rechtschreibfehler korrigieren.

Wenn der Text einer Debconf-Schablone korrigiert wurde und Sie **sicher** sind, dass die Änderung **keine** Übersetzungen beeinflusst, seien Sie so nett zu Übersetzern, die *Fuzzy*-Markierungen aus deren Übersetzungen zu entfernen.

Falls Sie dies nicht tun, wird die ganze Schablone nicht übersetzt, bis Ihnen ein Übersetzer eine Aktualisierung zusendet.

Um die *Fuzzy*-Markierungen aus Übersetzungen zu entfernen, können Sie **msguntypot** benutzen (Teil des Pakets `po4a`).

1. Erzeugen Sie die POT- und PO-Dateien neu.

   ```
   debconf-updatepo
   ```

2. Erstellen Sie eine Kopie der POT-Datei.

   ```
   cp templates.pot templates.pot.orig
   ```

3. Erstellen Sie eine Kopie aller PO-Dateien.

   ```
   mkdir po_fridge; cp *.po po_fridge
   ```

4. Ändern Sie die Debconf-Schablonen-Dateien, um den Tippfehler zu korrigieren.

5. Erzeugen Sie die POT- und PO-Dateien (wieder) neu.

   ```
   debconf-updatepo
   ```

   An dieser Stelle markiert die Korrektur des Tippfehlers alle Übersetzungen mit »fuzzy« und diese unglückliche Änderung ist die einzige zwischen den PO-Dateien Ihres Hauptverzeichnisses und den aus dem Kühlschrank. Hier nun eine Erklärung, wie das gelöst wird.

6. Verwerfen Sie die mit »fuzzy« markierte Übersetzung und stellen Sie die aus dem Kühlschrank wieder her.

   ```
   cp po_fridge/*.po .
   ```

7. Führen Sie manuell die PO-Dateien mit der neuen POT-Datei zusammen, aber berücksichtigen Sie das nutzlose »fuzzy«.

   ```
   msguntypot -o templates.pot.orig -n templates.pot *.po
   ```

8. Räumen Sie auf.

   ```
   rm -rf templates.pot.orig po_fridge
   ```

6.5.2.4 Treffen Sie keine Annahmen über Schnittstellen.

Schablonentext sollte keinen Bezug auf Steuerelemente herstellen, die zu irgendwelchen Debconf-Schnittstellen gehören. Sätze wie *If you answer Yes...* haben keinen Sinn für Benutzer grafischer Oberflächen, die für die Beantwortung logischer Fragen Kontrollkästchen verwenden.

Zeichenkettenschablonen sollten außerdem vermeiden, Vorgabewerte in ihrer Beschreibung zu erwähnen. Erstens sind diese zusätzlich zu den Werten, die der Anwender sieht, vorhanden. Außerdem könnten sich diese Werte von der Auswahl des Paketbetreuers unterscheiden (zum Beispiel, wenn die Debconf-Datenbank vorbelegt war).

Versuchen Sie, allgemein ausgedrückt, Bezug auf Benutzeraktionen zu vermeiden. Geben Sie nur Tatsachen wieder.

6.5.2.5 Reden Sie nicht in der ersten Person.

Sie sollten vermeiden, in der ersten Person zu reden (*I will do this...* oder *We recommend...*). Der Rechner ist keine Person und die Debconf-Schablonen sprechen nicht stellvertretend für Debian-Entwickler. Sie sollten neutrale Formulierungen benutzen. Diejenigen unter Ihnen, die bereits wissenschaftliche Publikationen verfasst haben, können ihre Schablonen so schreiben, als würden Sie wissenschaftliche Papiere verfassen. Versuchen Sie jedoch, wenn möglich, eine aktive Anrede zu verwenden, wie *Enable this if ...* anstelle von *This can be enabled if....*

6.5.2.6 Formulieren Sie geschlechtsneutral

Die Welt wird von Männern und Frauen bevölkert. Bitte benutzen Sie in Ihren Texten geschlechtsneutrale Formulierungen.

6.5.3 Definition von Schablonenfeldern

Dieser Teil stellt einige Informationen bereit, die meistens von der Handbuchseite debconf-devel(7) abgefragt wird.

6.5.3.1 Type

6.5.3.1.1 string resultiert in einem Eingabefeld freier Form, in das der Benutzer jegliche Zeichenkette eingeben kann.

6.5.3.1.2 password gibt dem Benutzer eine Eingabeaufforderung für ein Passwort aus. Benutzen Sie dies mit Vorsicht. Vergegenwärtigen Sie sich, dass das Passwort, das der Benutzer eingibt, in die Debconf-Datenbank geschrieben wird. Sie sollten diesen Wert möglicherweise aus der Datenbank löschen, sobald dies möglich ist.

6.5.3.1.3 boolean eine Auswahl »wahr/falsch«. Denken Sie daran: true/false, nicht **yes/no** ...

6.5.3.1.4 select Eine Auswahl aus mehreren Werten. Die Auswahlmöglichkeiten müssen in einem »Choices« benannten Feld angegeben werden. Trennen Sie die möglichen Werte mit Komma und Leerzeichen, wie hier: `Choices:yes, no, maybe`.

Falls Auswahlmöglichkeiten übersetzbare Zeichenketten sind, könnte das Feld durch Benutzung von `__Choices` als übersetzbar gekennzeichnet werden. Der doppelte Unterstrich wird jede Auswahl in eine separate Zeichenkette heraustrennen.

Das System **po-debconf** bietet außerdem interessante Möglichkeiten nur **einige** Auswahlmöglichkeiten als übersetzbar zu kennzeichnen. Ein Beispiel:

```
Template: foo/bar
Type: Select
#flag:translate:3
__Choices: PAL, SECAM, Other
_Description: TV standard:
 Please choose the TV standard used in your country.
```

In diesem Beispiel ist nur die Zeichenkette »Other« übersetzbar, während das andere Abkürzungen sind, die nicht übersetzt werden sollten. Obiges ermöglicht, dass nur »Other« in die POT- und PO-Dateien eingefügt wird.

Das Schaltersystem der Debconf-Schablonen bietet viele solcher Möglichkeiten. Die Handbuchseite po-debconf(7) führt all diese Möglichkeiten auf.

6.5.3.1.5 multiselect Wie der Datentyp »select«, außer dass der Benutzer eine beliebige Anzahl von Elementen aus der Auswahlliste auswählen kann (oder gar keins).

6.5.3.1.6 note Statt per se eine Frage zu sein, gibt dieser Datentyp eine Anmerkung an, die dem Benutzer angezeigt werden kann. Sie sollte nur für wichtige Anmerkungen benutzt werden, die der Benutzer wirklich sehen sollte, weil Debconf großen Aufwand betreibt, um sicherzustellen, dass der Benutzer sie sieht und die Installation anhält, so dass der Benutzer eine Taste drückt und sogar in manchen Fällen eine Benachrichtigung per E-Mail bekommt.

6.5.3.1.7 text Dieser Typ wird nun als veraltet angesehen: Benutzen Sie ihn nicht.

6.5.3.1.8 error Dieser Typ wurde entworfen, um Fehlermeldungen zu handhaben. Er ist meist dem Typ »note« ähnlich. Oberflächen könnten ihn unterschiedlich anzeigen (die Oberfläche von Cdebconf zeichnet beispielsweise einen roten statt des üblichen blauen Bildschirms).

Es wird empfohlen, diesen Typ für jegliche Nachricht zu verwenden, die die Aufmerksamkeit des Anwenders für irgendeine Art von Korrektur auf sich ziehen muss.

6.5.3.2 Description: Kurze und längere Beschreibung

Schablonenbeschreibungen haben zwei Teile: kurz und länger. Die Kurzbeschreibung steht in der Zeile »Description:« der Schablone.

Die Kurzbeschreibung sollte knapp gehalten werden (ungefähr 50 Zeichen), so dass sie in den meisten Debconf-Schnittstellen untergebracht werden kann. Es hilft obendrein Übersetzern, wenn sie kurz gehalten wird, da Übersetzungen normalerweise dazu neigen, länger als das Original zu sein.

Die Kurzbeschreibung sollte für sich allein stehen können. Einige Schnittstellen zeigen standardmäßig die ausführliche Beschreibung nicht, nur dann, wenn der Benutzer danach fragt oder sogar überhaupt nicht an. Vermeiden Sie Dinge wie »Was möchten Sie tun?«

Die Kurzbeschreibung muss nicht notwendigerweise aus einem vollständigen Satz bestehen. Dies ist Teil der Forderung nach kurzen, brauchbaren Empfehlungen.

Die längere Beschreibung sollte die Kurzbeschreibung nicht Wort für Wort wiederholen. Falls Ihnen keine ausführliche Beschreibung einfällt, denken Sie zuerst etwas mehr nach. Schreiben Sie an Debian-devel. Bitten Sie um Hilfe. Nehmen Sie Schreibunterricht! Diese längere Beschreibung ist wichtig. Falls Sie nach allem noch immer nicht damit zurecht kommen, lassen Sie sie leer.

Die längere Beschreibung sollte in ganzen Sätzen verfasst sein. Absätze sollten kurz gehalten werden, um die Leserlichkeit zu verbessern. Vermischen Sie nicht zwei Ideen in einem Absatz, sondern benutzen Sie lieber einen anderen Absatz.

Seien Sie nicht zu gesprächig. Benutzer tendieren dazu, zu lange Bildflächen zu ignorieren. 20 Zeilen sind erfahrungsgemäß die Grenze, die Sie nicht überschreiten sollten, da dies bedeuted, dass Anwender klassische Dialogfenster nicht scrollen müssen und viele Leute tun das einfach nicht.

Die längere Beschreibung sollte **keine** Frage enthalten.

Um etwas über besondere Regeln zu erfahren, die vom Schablonentyp (string, boolean etc.) abhängen, lesen Sie das Folgende.

6.5.3.3 Choices

Dieses Feld sollte für »select«- und »multiselect«-Typen verwandt werden. Es enthält die Auswahlmöglichkeiten, die dem Benutzer angezeigt werden. Diese Auswahlmöglichkeiten sollten durch Kommas getrennt werden.

6.5.3.4 Default

Dieses Feld ist optional. Es enthält die vorgegebene Antwort für die »string«-, »select«- und »multiselect«-Schablonen. Für »multiselect«-Schablonen könnte es eine durch Kommas getrennte Auswahlliste enthalten.

6.5.4 Stil-Anleitung speziell für Schablonenfelder

6.5.4.1 Feld »Type«

keine besondere Angabe, außer: Benutzen Sie den geeigneten Typ bezogen auf den vorhergehenden Abschnitt.

6.5.4.2 Feld »Description«

Es folgen spezifische Anweisungen für ordnungsgemäßes Verfassen der Beschreibung (kurz und länger), abhängig vom Schablonentyp.

6.5.4.2.1 »string«-/»password«-Schablonen

- Die Kurzbeschreibung ist eine Abfrage und **kein** Titel. Vermeiden Sie den Fragestil (IP address?) und geben Sie offenen Abfragen (IP address:) den Vorzug. Es wird empfohlen Doppelpunkte zu benutzen.
- Die längere Beschreibung ist eine Ergänzung der Kurzbeschreibung. Im erweiterten Teil erklären Sie, was gefragt ist, anstatt die gleiche Frage in längerer Formulierung wieder zu stellen. Benutzen Sie ganze Sätze. Von knappem Schreibstil wird strikt abgeraten.

6.5.4.2.2 »boolean«-Schablonen

- Die Kurzbeschreibung sollte in der Frageform ausgedrückt werden, die kurz gehalten und generell mit einem Fragezeichen beendet werden sollte. Knapper Schreibstil ist erlaubt und sogar gewollt, falls die Frage eher lang ist. (Denken Sie daran, dass Übersetzungen oft länger als die Originalversionen sind.)
- Nochmals: Bitte vermeiden Sie, sich auf Schnittstellen-spezifische Dinge zu beziehen. Es ist ein häufiges Missverständnis bei solchen Schablonen, wenn Sie Ja-Typ-Konstruktionen antworten.

6.5.4.2.3 »select«/»multiselect«

- Die Kurzbeschreibung ist eine Abfrage und **kein** Titel. Benutzen Sie **keine** nutzlosen »Please choose...«-Konstruktionen. Anwender sind klug genug herauszufinden, dass sie etwas auswählen sollen ... :)
- Die längere Beschreibung wird die Kurzbeschreibung vervollständigen. Sie könnte sich auf die verfügbaren Auswahlmöglichkeiten beziehen. Sie könnte zudem erwähnen, dass der Anwender unter mehr als einer verfügbaren Auswahlmöglichkeit wählen kann, falls es sich um eine »multiselect«-Schablone handelt.

6.5.4.2.4 »notes«

- Die Kurzbeschreigung sollte als **Titel** betrachtet werden.
- Die längere Beschreibung ist das, was als detaillierte Erklärung der Notiz angezeigt wird. Sätze, kein knapper Schreibstil.
- **Missbrauchen Sie Debconf nicht.** Notizen sind die häufigste Art, auf die Debconf missbraucht wird. Wie steht doch in der Handbuchseite von »debconf-devel« geschrieben: Es ist am Besten, dies nur für Warnungen über sehr ernsthafte Probleme zu benutzen. Die Dateien `NEWS.Debian` oder `README.Debian` sind für viele Notizen der passende Ort. Denken Sie, wenn Sie dies lesen, darüber nach, Ihre Schablonen des Typs »notes« zu Einträgen in `NEWS.Debian` oder `README.Debian` umzuwandeln und existierende Übersetzungen für die Zukunft aufzubewahren.

6.5.4.3 Das Feld »Choices«

Falls sich »Choices« zu oft ändert, sollten Sie in Betracht ziehen, zum __Choices-Trick zu greifen. Dies wird jede einzelne Auswahl in eine einzelne Zeichenkette aufteilen, was Übersetzern beträchtlich bei ihrer Arbeit helfen wird.

6.5.4.4 Das Feld »Default«

Falls der Vorgabewert für eine Auswahlschablone sich wahrscheinlich abhängig von der Sprache des Anwenders unterscheidet (zum Beispiel, weil es sich bei der Auswahl um eine Sprachauswahl handelt), benutzen Sie bitte den _Default-Trick.

Dieses Spezialfeld ermöglicht Übersetzern, die am Besten zu ihrer Sprache passende Auswahl zu nehmen. Es wird die vorgegebene Auswahl sein, wenn ihre Sprache benutzt wird, während Ihre eigene erwähnte »Default Choice« benutzt wird, wenn Sie Englisch benutzen.

Ein Beispiel aus den Schablonen des Pakets Geneweb:

```
Template: geneweb/lang
Type: select
__Choices: Afrikaans (af), Bulgarian (bg), Catalan (ca), Chinese (zh), Czech (cs) ↩
    , Danish (da), Dutch (nl), English (en), Esperanto (eo), Estonian (et),  ↩
    Finnish (fi), French (fr), German (de), Hebrew (he), Icelandic (is), Italian  ↩
    (it), Latvian (lv), Norwegian (no), Polish (pl), Portuguese (pt), Romanian ( ↩
    ro), Russian (ru), Spanish (es), Swedish (sv)
# This is the default choice. Translators may put their own language here
# instead of the default.
# WARNING : you MUST use the ENGLISH NAME of your language
# For instance, the french translator will need to put French (fr) here.
_Default: English[ translators, please see comment in PO files]
_Description: Geneweb default language:
```

Beachten Sie, dass die Benutzung von Klammern Kommentare in Debconf-Feldern erlaubt. Beachten Sie außerdem, dass die Benutzung von Kommentaren in Dateien zu sehen sein wird, mit denen Übersetzer arbeiten.

Die Kommentare werden benötigt, da der _Default-Trick etwas verwirrendes ist: Die Übersetzer könnten ihre eigene Auswahl nehmen.

6.5.4.5 Das Feld »Default«

Benutzen Sie KEIN leeres »Default«-Feld. Falls Sie keine Vorgabewerte benutzen möchten, benutzen Sie Default überhaupt nicht.

Falls Sie Po-debconf benutzen (und das **sollten** Sie, lesen Sie Abschnitt 6.5.2.2), erwägen Sie, dieses Feld übersetzbar zu machen, wenn Sie der Meinung sind, es könnte übersetzt werden.

Falls der Vorgabewert von Sprache oder Land abhängen könnte (zum Beispiel, weil es sich bei der Auswahl um eine Sprachauswahl handelt), ziehen Sie in Betracht, den Typ _Default zu benutzen, der in po-debconf(7) dokumentiert wird.

6.6 Internationalisierung

Der zweite Abschnitt enthält globale Informationen für Entwickler, um Übersetzern das Leben leichter zu machen. Weitere Informationen für Übersetzer und Entwickler, die sich für Internationalisierung interessieren, sind in der Dokumentation Internationalisierung und Lokalisierung verfügbar.

6.6.1 Handhabung von Debconf-Übersetzungen

Wie Portierende haben auch Übersetzer noch andere Aufgaben. Sie arbeiten an vielen Paketen und müssen mit vielen verschiedenen Paketbetreuern zusammenwirken. Außerdem ist Englisch meist nicht ihre Muttersprache. Sie sollten ihnen daher besondere Geduld entgegenbringen.

Das Ziel von `debconf` war die Vereinfachung der Paketkonfiguration für Betreuer und Anwender. Ursprünglich wurde die Übersetzung von Debconf-Schablonen mit **debconf-mergetemplate** gehandhabt. Nun wird diese Technik jedoch missbilligt. Die Internationalisierung von `debconf` ist am besten mit dem Paket `po-debconf` zu erreichen. Diese Methode ist sowohl für Betreuer als auch für Übersetzer einfacher. Es werden Umwandlungsskripte bereitgestellt.

Wenn `po-debconf` benutzt wird, werden sie Übersetzungen in `.po`-Dateien gespeichert (mit **gettext**-Übersetzungstechniken herausgezogen). Spezielle Schablonendateien enthalten die Originalnachrichten und markieren, welche Felder übersetzbar sind. Wenn Sie den Wert eines übersetzbaren Feldes durch Aufruf von **debconf-updatepo** ändern, wird die Übersetzung für Übersetzerals aufmerksamkeitsbedürfend gekennzeichnet. Dann, zur Build-Zeit, wird das Programm **dh_installdebconf** wie von Zauberhand dafür sorgen, dass alle Schablonen zusammen mit den aktuellen Übersetzungen in die Binärpakete einfließen. Weitere Einzelheiten können Sie der Handbuchseite po-debconf(7) entnehmen.

6.6.2 Internationalisierte Dokumentation

Internationalisierte Dokumentation für Anwender ist wichtig, bereitet aber viel Mühe. Es gibt keine Möglichkeit, all diese Arbeit zu beseitigen, aber Sie können den Übersetzern einige Dinge erleichtern.

Falls Sie Dokumentationen in irgendwelchem Umfang betreuen, ist es für Übersetzer einfacher, wenn Sie Zugriff auf das Versionsverwaltungssystem haben. Dadurch können Übersetzer die Unterschiede zwischen zwei Versionen der Dokumentation anschauen, so dass sie beispielsweise sehen können, was neu übersetzt werden muss. Es wird empfohlen, dass die übersetzte Dokumentation eine Notiz darüber bereithält, auf welcher Revision der Quellenverwaltung die Übersetzung basiert. Ein interessantes System wird von doc-check aus dem `debian-installer`-Paket bereitgestellt, das eine Übersicht über den Übersetzungsstatus für eine angegebene Sprache anzeigt. Dazu werden strukturierte Kommentare für die aktuelle Revision der zu übersetzenden Datei und für eine übersetzte Datei die Revision des Originals auf der die Übersetzung basiert, angezeigt. Möglicherweise möchten Sie dies anpassen und in Ihrem VCS-Bereich bereitstellen.

Falls Sie XML- oder SGML-Dokumentationen betreuen, wird geraten, dass Sie jegliche sprachabhängigen Informationen isolieren und diese als Instanzen in einer separaten Datei definieren, die in allen verschiedenen Übersetzungen enthalten ist. Dies macht es beispielsweise viel einfacher, URLs über mehrere Dateien hinweg aktuell zu halten.

Einige Werkzeuge (z.B. `po4a`, `poxml` oder `translate-toolkit`) sind darauf spezialisiert, übersetzbares Material aus verschiedenen Formaten zu extrahieren. Sie erstellen PO-Dateien, ein für Übersetzer ziemlich häufiges Format, das eine Übersicht darüber gibt, was übersetzt werden muss, wenn das übersetzte Dokument aktualisiert wurde.

6.7 Übliche Paketierungssituationen

6.7.1 Pakete benutzen autoconf/automake

Die Dateien `config.sub` und `config.guess` von **autoconf** aktuell zu halten ist für Portierende kritisch, insbesondere auf eher unbeständigen Architekturen. Einige sehr gute Paketierungsvorgehensweisen für irgendwelche Pakete, die **autoconf** und/oder **automake** benutzen, wurden in `/usr/share/doc/autotools-dev/README.Debian.gz` aus dem Paket `autotools-dev` zusammengefasst. Es wird eindringlich geraten, diese Datei und die folgenden Empfehlungen zu lesen.

6.7.2 Bibliotheken

Bibliotheken unterscheiden sich immer von Paketen aus unterschiedlichen Gründen. Die Richtlinien verhängen mehrere Beschränkungen, um ihre Verwaltung zu erleichtern und sicherzustellen, dass Upgrades so einfach wie möglich sind, wenn eine neue Originalversion herauskommt. Eine kaputte Bibliothek kann dazu führen, dass Dutzende davon abhängige Pakete kaputtgehen.

Gute Vorgehensweisen für das Paketieren von Bibliotheken wurden in der Anleitung zum Paketieren von Bibliotheken zusammengefasst.

6.7.3 Dokumentation

Achten Sie darauf, dass Sie den Richtlinien für Dokumentation folgen.

Falls Ihr Paket Dokumentation enthält, die aus XML oder SGML erstellt wurde, wird empfohlen, nicht die XML- oder SGML-Quellen im (in den) Binärpaket(en) mitzuliefern. Falls Anwender die Quelle der Dokumentation möchten, sollten sie die Paketquelle abrufen.

Die Richtlinie gibt an, dass die Dokumentation im HTML-Format weitergegeben werden sollte. Außerdem wird empfohlen, die Dokumentation im PDF-Format und als Klartext mitzuliefern, falls geeignet und falls die Ausgabe in einer vernünftigen Qualität möglich ist. Es ist allgemein jedoch nicht angemessen, Klartextversionen von Dokumentationen mitzuliefern, deren Quellformat HTML ist.

Bedeutende mitgelieferte Handbücher sollten sich selbst bei der Installation mit `doc-base` registrieren. Weitere Einzelheiten erhalten Sie in der Dokumentation des Pakets `doc-base`.

Die Debian-Richtlinien (Abschnitt 12.1) schreiben vor, dass Handbuchseiten jedes Programm, jedes Hilfswerkzeug und jede Funktion begleiten sollten und für andere Objekte, wie Konfigurationsdateien, nahegelegt werden. Falls die von Ihnen paketierte Arbeit nicht über eine solche Handbuchseite verfügt, dann überlegen Sie sich, eine zu schreiben, die Ihrem Paket beigefügt und an die Originalautoren gesandt wird.

Die Handbuchseiten müssen nicht direkt im Troff-Format geschrieben werden. Beliebte Quellformate sind Docbook, POD und reST, die mit **xsltproc**, **pod2man** beziehungsweise **rst2man** umgewandelt werden können. In geringerem Maße kann außerdem das Programm **help2man** benutzt werden, um einen Abschnitt zu schreiben.

6.7.4 Besondere Pakettypen

Mehrere besondere Typen von Paketen haben spezielle Unterrichtlinien und zugehörige Paketierungsregeln und -Vorgehensweisen:

- Perl zugehörige Pakete haben eine Perl-Richtlinie. Einige Beispiele für Pakete, die dieser Richtlinie folgen, sind `libdbd-pg-perl` (binäres Perl-Modul) oder `libmldbm-perl` (architekturunabhängiges Perl-Modul).
- Python zugehörige Pakete haben ihre Python-Richtlinie. Siehe `/usr/share/doc/python/python-policy.txt.gz` im Paket `python`.
- Emacs zugehörige Pakete haben die Emacs-Richtlinie.
- Java zugehörige Pakete haben ihre Java-Richtlinie.
- Ocaml zugehörige Pakete haben ihre eigene Richtlinie, die unter `/usr/share/doc/ocaml/ocaml_packaging_policy.gz` im Paket `ocaml` gefunden werden kann. Ein gutes Beispiel ist das Quellpaket `camlzip`.
- Pakete, die XML- oder SGML-DTDs bereitstellen, sollten konform zu den Empfehlungen in Paket `sgml-base-doc` sein.
- Lisp-Pakete sollten sich selbst mit `common-lisp-controller` registrieren. Siehe dazu `/usr/share/doc/common-lisp-controller/README.packaging`.

6.7.5 Architekturunabhängige Daten

Es ist nicht unüblich, eine größere Menge architekturunabhängiger Daten mit einem Programm zu paketieren, Zum Beispiel Audiodateien, eine Symbolsammlung, Hintergrundmuster oder grafische Dateien. Falls die Größe dieser Daten vernachlässigbar im Vergleich zum Rest des Pakets ist, ist es wahrscheinlich am besten, alles in einem einzelnen Paket zu halten.

Ist die Größe allerdings beachtlich, denken Sie darüber nach, sie in ein separates architekturunabhängiges Paket (`_all.deb`) auszulagern. Imdem Sie dies tun, vermeiden Sie nutzlose Vervielfältigung der gleichen Daten in elf oder mehr .debs, eins je Architektur. Indem dies einigen zusätzlichen Zuschlag zu den `Packages`-Dateien hinzufügt, spart es viel Plattenplatz auf Debian-Spiegeln. Das Heraustrennen architekturunabhängiger Daten vermindert auch die Ausführungszeit von **lintian** (siehe Abschnitt A.2), wenn es für das ganze Debian-Archiv ausgeführt wird.

6.7.6 Eine bestimmte Locale wird während des Builds benötigt

Falls Sie eine bestimmte Locale während des Builds benötigen, können Sie mittels dieses Tricks eine temporäre Datei erstellen:

Falls Sie `LOCPATH` auf die Entsprechung von `/usr/lib/locale` und `LC_ALL` auf den Namen der Locale setzen, die sie generieren, sollten Sie erreichen, was Sie möchten ohne dass Sie Root sind. Etwas wie Folgendes:

```
LOCALE_PATH=debian/tmpdir/usr/lib/locale
LOCALE_NAME=en_IN
LOCALE_CHARSET=UTF-8

mkdir -p $LOCALE_PATH
localedef -i $LOCALE_NAME.$LOCALE_CHARSET -f $LOCALE_CHARSET $LOCALE_PATH/ ↩
    $LOCALE_NAME.$LOCALE_CHARSET

# Using the locale
LOCPATH=$LOCALE_PATH LC_ALL=$LOCALE_NAME.$LOCALE_CHARSET date
```

6.7.7 Machen Sie vorübergehende Pakete Deborphan-konform

Deborphan ist ein Programm, das Anwendern hilft Pakete aufzuspüren, die sicher vom System entfernt werden können, d.h. diejenigen, von denen keine Pakete abhängen. Die Standardoperation ist, nur innerhalb der Abschnitte »libs« und »oldlibs« zu suchen, um Jagd auf unbenutzte Bibliotheken zu machen. Wenn aber das richtige Argument übergeben wird, versucht es auch andere nutzlose Pakete zu erwischen.

Mit `--guess-dummy` versucht **deborphan** zum Beispiel alle vorübergehenden Pakete zu suchen, die zum Upgrade benötigt wurden, die nun aber sicher entfernt werden können. Dazu sucht es nach den Zeichenketten »dummy« oder »transitional« in dessen Kurzbeschreibung.

Wenn Sie also solch ein Paket erstellen, achten Sie bitte darauf, diesen Text zu Ihrer Kurzbeschreibung hinzuzufügen. Falls Sie sich nach Beispielen umsehen, führen Sie einfach **apt-cache search .|grep dummy** oder **apt-cache search .|grep transitional** aus.

Außerdem wird empfohlen, den Abschnitt in `oldlibs` und die Priorität in `extra` anzupassen, um die Arbeit von **deborphan** zu erleichtern.

6.7.8 Optimale Vorgehensweisen für `.orig.tar.{gz,bz2,xz}`-Dateien

Es gibt zwei Arten von Original-Quell-Tarballs: unberührte Quellen und neu paketierte Quellen der Originalautoren.

6.7.8.1 Unberührte Quellen

Das charakteristische Merkmal eines unberührten Tarballs ist, dass die `.orig.tar.{gz,bz2,xz}`-Datei Byte für Byte identisch mit einem offiziell weitergegebenen Tarball des Originalautors ist.[1] Dies ermöglicht die Benutzung von Prüfsummen, um auf einfache Weise alle Änderungen zwischen Debians Version und der der Originalautoren zu prüfen, die in der Diff-Datei in Debian enthalten sind. Falls außerdem die Originalquelle riesig ist, können Originalautoren und andere, die bereits den Original-Tarball haben, Download-Zeit sparen, falls sie Ihre Paketierung im Detail inspizieren möchten.

Es gibt keine allgemein anerkannten Richtlinien, denen Originalautoren betreffend der Verzeichnisstruktur innerhalb ihres Tarballs folgen, aber **dpkg-source** ist dennoch in der Lage mit den meisten Tarballs von Originalautoren als unberührte Quelle umzugehen. Seine Strategie entspricht dem Folgenden:

1. Es entpackt den Tarball in eine leeres temporäres Verzeichnis mittels

   ```
   zcat path/to/Paketname_Originalversion.orig.tar.gz | tar xf -
   ```

2. Falls das temporäre Verzeichnis danach nur ein Verzeichnis und keine anderen Dateien enthält, benennt **dpkg-source** dieses Verzeichnis in *`Paketname_Originalversion`*`(.orig)` um. Der Name des Verzeichnisses auf der obersten Ebene im Tarball ist ohne Bedeutung und geht verloren.

3. Andernfalls muss der Tarball der Originalautoren ohne ein sonst übliches Verzeichnis der obersten Ebene gepackt worden sein (Schande über den Originalautor!). In diesem Fall benennt **dpkg-source** das temporäre Verzeichnis *selbst* in *`Paketname_Originalversion`*`(.orig)` um.

6.7.8.2 Neu paketierte Originalquelle

Sie **sollten** Pakete, wenn möglich, mit einem unberührten Quell-Tarball hochladen, aber es gibt viele Gründe, warum das manchmal nicht möglich ist. Dies ist der Fall, wenn die Originalautoren die Quelle gar nicht als Gzip-gepackte Tar-Datei weitergeben oder falls der Tarball der Originalautoren nicht DFSG-freies Material enthält, das Sie vor den Hochladen entfernen müssen.

In diesen Fällen muss der Entwickler selbst eine geeignete `.orig.tar.{gz,bz2,xz}`-Datei bauen. Solch ein Tarball wird neu paketierten Originalquellen zugeordnet. Beachten Sie, dass sich eine neu paketierte Originalquelle von einem nativen Debian-Paket unterscheidet. Eine neu paketierte Quelle kommt mit Debian-spezifischen Änderungen in einem separaten `.diff.gz` oder `.debian.tar.{gz,bz2,xz}` daher und hat eine Versionsnummer, die sich aus der *`Originalversion`* und der *`Debian-version`* zusammensetzt.

Es könnte Gründe geben, aus denen es wünschenswert wäre, die Quelle neu zu paketieren, obwohl die Originalautoren ein `.tar.{gz,bz2,xz}` verteilen, dass im Prinzip in seiner unberührten Form benutzt werden könnte. Der naheliegendste Grund ist, wenn *signifikante* Platzersparnis durch Neukomprimierung des Tar-Archivs oder Entfernen von wirklich nutzlosem Müll aus dem Qriginalarchiv erzielt werden kann. Handeln Sie hier nach eigenem Ermessen, aber seien Sie darauf vorbereitet, Ihre Entscheidung zu verteidigen, falls Sie eine Quelle neu paketieren, die unberührt sein könnte.

Ein neu paketiertes `.orig.tar.{gz,bz2,xz}`

[1] Originalautoren können nicht daran gehindert werden, den Tarball, den sie verteilen, zu ändern ohne die Versionsnummer zu erhöhen, daher kann nicht gewährleistet werden, dass ein unberührter Tarball mit dem identisch ist, was die Originalautoren *aktuell* zu irgendeinem Zeitpunkt weitergeben. Alles was erwartet werden kann, ist, dass es identisch ist mit etwas ist, das die Originalautoren einmal weitergegeben *haben*. Falls sich später ein Unterschied ergibt (etwa, wenn die Originalautoren merken, dass sie in ihrer Distribution des Originals keine maximale Komprimierung nutzen und es dann erneut mit **gzip** packen), ist das einfach Pech. Da es keine brauchbare Möglichkeit gibt, ein neues `.orig.tar.{gz,bz2,xz}` für die gleiche Version hochzuladen, gibt es auch keinen Punkt, an dem diese Situation als ein Fehler behandelt wird.

1. **sollte** im resultierenden Quellpaket dokumentiert sein. Detaillierte Informationen, wie die neu paketierte Quelle gewonnen wurde und wie dies reproduziert werden kann, sollten in `debian/copyright` bereitgestellt werden. Es ist außerdem eine gute Idee, ein `get-orig-source`-Target in Ihrer `debian/rules`-Datei bereitzustellen, die den Prozess wiederholt, wie im Richtlinien-Handbuch beschrieben Main building script: `debian/rules`.

2. **sollte keine** Datei enthalten, die nicht von dem/den Origianlautor(en) stammt oder deren Inhalt von Ihnen geändert wurde.[2]

3. **sollte** außer, wenn es aus rechtlichen Gründen unmöglich ist, die ganze Erstellungs- und Portierungsinfrastruktur aufbewahren, die vom Originalautor bereitgestellt wurde. Es ist zum Beispiel kein ausreichender Grund für das Weglassen einer Datei, wenn sie nur für die Erstellung unter MS-DOS benutzt wird. Gleichermaßen sollte ein `Makefile`, das vom Originalautor bereitgestellt wurde nicht einmal dann weggelassen werden, wenn das erste, was Ihre `debian/rules` tut, das Überschreiben durch Ausführen eines Konfigurationsskripts ist.

   (*Begründung:* Es ist üblich für Debian-Anwender, die Software für nicht-Debian-Plattformen erstellen möchten, die Quellen von einem Debian-Spiegel abzurufen, anstatt den Punkt der ordnungsgemäßen Originaldistribution zu suchen).

4. **sollte** als Namen des Verzeichnisses auf der obersten Ebene des Tarballs *`Paketname-Originalversion`*`.orig` benutzen. Dies ermöglicht die Unterscheidung von unberührten und neu paketierten Tarballs.

5. **sollte** mit Gzip oder Bzip mit der maximalen Komprimierung gepackt werden.

6.7.8.3 Ändern binärer Dateien

Manchmal ist es nötig, binäre Dateien zu ändern, die im Original-Tarball enthalten sind oder binäre Dateien hinzuzufügen, die nicht darin enthalten sind. Dies wird vollständig unterstützt, wenn Sie Quellpakete im Format »3.0 (quilt)« benutzen. Lesen Sie die Handbuchseite dpkg-source(1), um weitere Einzelheiten zu erfahren. Wenn Sie das ältere Format »1.0« benutzen, können binäre Dateien nicht im `.diff.gz` gespeichert werden, daher müssen Sie eine mit **uuencode** (oder ähnlichem) kodierte Version der Datei(en) speichern und zur Erstellungszeit in `debian/rules` entschlüsseln (und an ihren offiziellen Platz verschieben).

6.7.9 Optimale Vorgehensweisen für Debug-Pakete

Ein Debug-Paket ist ein Paket, dessen Name mit -dbg endet. Es enthält zusätzliche Informationen, die **gdb** benutzen kann. Da Debian-Programme standardmäßig unverhüllt sind, sind Debugging-Informationen, einschließlich Namen und Zeilennummern andernfalls nicht verfügbar, wenn **gdb** auf Debian-Programmen ausgeführt wird. Debug-Pakete ermöglichen Anwendern, die diese zusätzlichen Debugging-Informationen benötigen, sie zu installieren ohne das normale System mit diesen Informationen aufzublähen.

Es liegt beim Paketbetreuer, ob ein Debug-Paket erstellt wird oder nicht. Betreuer werden aufgefordert Debug-Pakete für Bibliothekenpakete zu erstellen, da dies bei der Fehlersuche in vielen Programmen helfen kann, die mit der Bibliothek verlinkt werden. Im Allgemeinen gibt es keine Notwendigkeit für Debug-Pakete zu allen Programmen; dies würde das Archiv aufblähen. Falls aber ein Betreuer findet, dass Anwender oft die Debugging-Version eines Programms benötigen, kann es lohnenswert sein, ein Debug-Paket dafür zu erstellen. Programme die zur Kerninfrastruktur gehören, wie Apache oder der X-Server sind ebenfalls geeignete Kandidaten für Debug-Pakete.

Einige Debug-Pakete könnten ein ganz spezielles Debugging-Build einer Bibliothek oder eines anderen Programms haben, aber die meisten können Speicher und Build-Zeit sparen, indem sie stattdessen separate Debugging-Symbole enthalten, die **gdb** spontan finden und laden kann, wenn in einem Programm oder einer Bibliothek nach Fehlern gesucht wird. Die Konvention in Debian besagt, dass diese Symbole in `/usr/lib/debug/`*`Pfad`* aufbewahrt werden, wobei *`Pfad`* der Pfad zum ausführbaren Programm oder der Bibliothek ist. Debugging-Symbole für `/usr/bin/foo` wandern beispielsweise nach `/usr/lib/debug/usr/bin/foo` und Debugging-Symbole für `/usr/lib/libfoo.so.1` nach `/usr/lib/debug/usr/lib/libfoo.so.1`.

Die Debugging-Symbole können mit **objcopy --only-keep-debug** aus einer Objektdatei extrahiert werden. Dann kann die Objektdatei enthüllt und **objcopy --add-gnu-debuglink** benutzt werden, um den Pfad zur Debugging-Symboldatei anzugeben. objcopy(1) erklärt im Detail, wie dies funktioniert.

[2] Als besondere Ausnahme könnte, falls das Auslassen unfreier Dateien dazu führen würde, dass das Build der Quelle ohne Unterstützung aus dem Debian-Diff fehlschlägt, das Bearbeiten der Dateien anstelle des Weglassens unfreier Teile davon und/oder Erklären der Situation in einer `README.source`-Datei im Wurzelverzeichnis der Quelle angemessen sein. Ermahnen Sie in diesem Fall aber auch den Originalautor, unfreie Komponenten leichter aus dem Quelltext heraustrennbar zu machen.

Der Befehl **dh_strip** in `debhelper` unterstützt das Erstellen von Debug-Paketen und kann sich um die Benutzung von **objcopy** kümmern, um die Debugging-Symbole für Sie herauszusuchen. Falls Ihr Paket `debhelper` benutzt, müssen Sie nur **dh_strip --dbg-package=libfoo-dbg** aufrufen und einen Eintrag in `debian/control` für das Debug-Paket hinzufügen.

Beachten Sie, dass Debug-Pakete von dem Paket abhängen sollten, für das sie Debugging-Symbole bereitstellen und diese Abhängigkeit sollte mit einer Version versehen werden. Zum Beispiel:

```
Depends: libfoo (= ${binary:Version})
```

6.7.10 Optimale Vorgehensweisen für Meta-Pakete

Ein Meta-Paket ist meist ein leeres Paket, das es vereinfacht, eine Zusammenstellung von Paketen zu installieren, die sich im Lauf der Zeit weiterentwickeln können. Es erreicht dies, indem es von allen Paketen der Zusammenstellung abhängt. Dank der Fähigkeiten von APT kann der Betreuer des Meta-Pakets die Abhängigkeiten anpassen und das System des Anwenders wird automatisch die zusätzlichen Pakete erhalten. Die weggelassenen Pakete, die automatisch installiert wurden, werden außerdem als Kandidaten für das Entfernen gekennzeichnet (und werden sogar durch **aptitude** automatisch entfernt). `gnome` und `linux-image-amd64` sind zwei Beispiele für Meta-Pakete (gebaut durch die Quellpakete `meta-gnome2` und `linux-latest`).

Die ausführliche Beschreibung des Meta-Pakets muss ihren Zweck klar dokumentieren, so dass der Benutzer weiß, was er verliert, wenn er das Paket entfernt. Es wird empfohlen, genau über die Konsequenzen zu informieren. Dies ist besonders für Meta-Pakete wichtig, die während der anfänglichen Installation installiert werden und nicht explizit durch den Benutzer installiert wurden. Diese neigen dazu, wichtig für reibungslose Upgrades des Systems zu sein und der Benutzer sollte entmutigt werden, sie zu entfernen, um mögliche Schäden zu vermeiden.

Kapitel 7

Jenseits der Paketierung

Debian ist weit mehr als nur Software zu paketieren und diese Pakete zu verwalten. Dieses Kapitel enthält Informationen über Wege, oft wirklich kritische Wege, fernab von einfachem Erstellen und Verwalten von Paketen zu Debian beizutragen.

Als eine Organisation von Freiwilligen verlässt sich Debian auf das Ermessen seiner Mitglieder, die auswählen, an was sie arbeiten möchten und was die kritischste Sache ist, in die sie Zeit investieren.

7.1 Fehler berichten

Bitte reichen Sie Fehlerberichte ein, wenn Sie Fehler in Debian-Paketen finden. Tatsächlich bilden Debian-Entwickler oftmals die erste Reihe der Tester. Fehler in den Paketen anderer Entwickler zu finden und zu melden, erhöht die Qualität von Debian.

Lesen Sie die Anweisungen zum Melden von Fehlern in der Debian-Fehlerdatenbank.

Versuchen Sie, den Fehlerbericht von einem normalen Benutzerkonto zu senden, von dem Sie wahrscheinlich Mails empfangen können, so dass Leute, die weitere Informationen über den Fehler benötigen, Sie erreichen können. Senden Sie keine Fehlerberichte als Root.

Sie können ein Werkzeug wie reportbug(1) benutzen, um Fehlerberichte zu senden. Es kann den Prozess automatisieren und generell erleichtern.

Stellen Sie sicher, dass der Fehler nicht bereits gegen das Paket eingereicht wurde. Jedes Paket hat eine Fehlerliste, die einfach unter `http://bugs.debian.org/`*`Paketname`* einsehbar ist. Hilfswerkzeuge wie querybts(1) können Sie ebenfalls mit diesen Informationen versorgen (und **reportbug** wird normalerweise auch vor dem Versenden **querybts** aufrufen).

Versuchen Sie, Ihre Fehlerberichte an die ordnungsgemäße Stelle zu lenken. Wenn Ihr Fehlerbericht zum Beispiel von einem Paket handelt, das Dateien eines anderen Pakets überschreibt, prüfen Sie die Fehlerliste *beider* Pakete, um das Einreichen doppelter Fehlerberichte zu vermeiden.

Um zusätzliche Anerkennung zu erhalten, vereinigen Sie Fehler, die mehr als einmal berichtet wurden oder kennzeichnen Sie Fehler als »fixed«, die bereits behoben wurden. Beachten Sie, dass wenn Sie weder der Absender des Fehlers noch der Betreuer des Pakets sind, den Fehlerbericht nicht tatsächlich schließen sollten (es sei denn, Sie versichern sich der Erlaubnis des Betreuers).

Von Zeit zu Zeit möchten Sie vielleicht prüfen, was aus den Fehlerberichten wurde, die Sie versandt haben. Nutzen Sie diese Gelegenheit, um diejenigen zu schließen, die Sie nicht mehr reproduzieren können. Um herauszufinden, welche Fehlerberichte Sie versandt haben, besuchen Sie `http://bugs.debian.org/from:`*`Ihre-E-Mail-Adresse`*.

7.1.1 Viele Fehler auf einmal berichten (Masseneinreichung von Fehlern)

Das Berichten einer großen Anzahl von Fehlern für dasselbe Problem für eine große Zahl von Paketen — d.h. mehr als zehn — ist eine missbilligte Vorgehensweise. Unternehmen Sie alle nötigen Schritte, um das massenhafte Versenden von Fehlerberichten tunlichst zu vermeiden. Prüfen Sie zum Beispiel, ob das Problem automatisiert werden kann, indem Sie eine neue Prüfung zu `lintian` hinzufügen, so dass eine Fehlermeldung oder Warnung ausgegeben wird.

Falls Sie mehr als zehn Fehler auf einmal zum gleichen Thema berichten, wird empfohlen, dass Sie eine Nachricht an debian-devel@lists.debian.org senden, in der Sie Ihre Absicht darlegen, ehe Sie den Bericht versenden und die Tatsache im Betreff Ihrer Mail erwähnen. Dies wird anderen Entwicklern ermöglichen zu prüfen, ob der

Fehler ein echtes Problem darstellt. Zusätzlich wird es helfen eine Situation zu vermeiden, in der mehrere Betreuer simultan beginnen, den gleichen Fehlerbericht einzureichen.

Bitte benutzen Sie das Programm **dd-list** und falls geeignet **whodepends** (aus dem Paket `devscripts`), um eine Liste aller betroffenen Pakete zu generieren und fügen sie die Ausgabe in Ihre Mail an debian-devel@lists.debian.org ein.

Beachten Sie, wenn Sie viele Fehlerberichte mit dem gleichen Betreff senden möchten, dass Sie den Fehlerbericht an maintonly@bugs.debian.org senden sollten, so dass der Fehlerbericht nicht an die Fehlerverteilungs-Maillingliste weitergeleitet wird.

7.1.1.1 Benutzerkennzeichen

Vielleicht möchten Sie BTS-Benutzerkennzeichen benutzen, wenn Sie Fehlerberichte über eine größere Anzahl Pakete senden. Benutzerkennzeichen sind normalen Kennzeichen wie »patch« oder »wishlist« ähnlich, unterscheiden sich aber darin, dass sie benutzerdefiniert sind und einen Namensraum belegen, der einzigartig für einen bestimmten Benutzer ist. Dies ermöglicht mehreren Gruppierungen von Entwicklern, den gleichen Fehler benutzerdefiniert auf unterschiedliche Arten zu kennzeichnen ohne Konflikte zu verursachen.

Um beim Einreichen von Fehlerberichten Benutzerkennzeichen hinzuzufügen, geben Sie die Pseudokopfzeilen `User` und `Usertags` an:

```
To: submit@bugs.debian.org
Subject: Titel des Fehlerberichts

Package: Paketname
[ ... ]
User: E-Mail-Adresse
Usertags: Kennzeichen [ Kennzeichen ... ]

Beschreibung des Fehlers ...
```

Beachten Sie, dass Kennzeichnungen durch Leerzeichen getrennt werden und keine Unterstriche enthalten dürfen. Falls Sie Fehlerberichte für eine spezielle Gruppe oder ein Team einreichen, wird empfohlen, dass Sie `User` auf eine angemessene Mailingliste setzen, nachdem Sie Ihre Absicht dort geschildert haben.

Um Fehler mit einem bestimmten Benutzerkennzeichen anzusehen, besuchen Sie `http://bugs.debian.org/cgi-bin/pkgreport.cgi?users=`*`E-Mail-Adresse`*`&tag=`*`Kennzeichen`*.

7.2 Qualitätssicherungsbestreben

7.2.1 Tägliche Arbeit

Auch wenn es eine eigene Gruppe zur Qualitätssicherung gibt, sind QS-Pflichten nicht ausschließlich dieser Gruppe vorbehalten. Sie können an dieser Aufgabe teilnehmen, indem Sie Ihre Pakete so fehlerfrei und so Lintian-rein (siehe Abschnitt A.2.1) wie möglich halten. Falls Sie finden, dies sei unmöglich, dann sollten Sie darüber nachdenken, einige Ihrer Pakete zu verwaisen (siehe Abschnitt 5.9.4). Alternativ könnten Sie andere Leute um Hilfe bitten, um den Rückstand, den Sie bei den Fehlern haben, aufzuholen (Sie können auf debian-qa@lists.debian.org oder debian-devel@lists.debian.org nach Hilfe fragen). Gleichzeitig können Sie sich nach Mitbetreuern umsehen (siehe Abschnitt 5.12).

7.2.2 Bug-Squashing-Parties

Von Zeit zu Zeit organisiert die QS-Gruppe Bug-Squashing-Parties, um so viele Probleme wie möglich zu beseitigen. Sie werden auf debian-devel-announce@lists.debian.org angekündigt und in der Ankündigung wird erklärt, auf welchem Bereich der Fokus der Party liegt: Üblicherweise liegt der Fokus auf veröffentlichungskritischen Fehlern, aber es kann vorkommen, dass entschieden wird, bei der Fertigstellung eines Haupt-Upgrades zu helfen (wie einer neuen **perl**-Version, die ein Neukompilieren aller binären Module erfordert).

Die Regeln für Non-Maintainer-Uploads unterscheiden sich während der Parties, da die Ankündigung der Party als vorausgehende Ankündigung für den NMU angesehen wird. Falls Sie Pakete haben, die von der Party betroffen sind (da sie zum Beispiel veröffentlichungskritische Fehler enthalten), sollten Sie eine Aktualisierung zu jedem zugehörigen Fehler senden, um seinen aktuellen Status zu erklären und und was Sie von der Party erwarten. Falls Sie keinen NMU möchten, nicht an einem Patch interessiert sind oder den Fehler selbst bewältigen möchten, erklären Sie dies bitte im BTS.

Teilnehmer der Party haben besondere Regeln für NMUs. Sie können NMUs ohne vorherige Ankündigung durchführen, falls sie mindestens nach DELAYED/3-day hochladen. Alle anderen NMU-Regeln gelten wie üblich; sie sollten den Patch des NMUs an das BTS senden (an einen der offenen Fehler, der durch den NMU behoben wird oder an einen neuen Fehler, der als »fixed« gekennzeichnet wird). Sie sollten außerdem die besonderen Wünsche des Betreuers respektieren.

Falls Sie sich nicht sicher fühlen, einen NMU durchzuführen, senden Sie nur einen Patch an das BTS. Das ist weit besser als ein kaputter NMU.

7.3 Andere Paketbetreuer kontaktieren

Während Ihres Lebens innerhalb Debian werden Sie aus verschiedenen Gründen Kontakt zu anderen Betreuern haben. Sie möchten vielleicht neue Wege der Kooperation zwischen einer Zusammenstellung verwandter Pakete diskutieren oder einfach jemanden daran erinnern, dass eine neue Originalversion verfügbar ist, die Sie benötigen.

Die E-Mail-Adresse des Paketbetreuers herauszusuchen, kann störend sein. Glücklicherweise gibt es einen einfachen E-Mail-Alias, *Paket*`@packages.debian.org`, der eine Möglichkeit bietet, dem Betreuer zu mailen, ganz gleich wie seine E-Mail-Adresse (oder Adressen) auch sein mag. Ersetzen Sie *Paket* durch den Namen eines Quell- oder Binärpakets.

Möglicherweise sind Sie außerdem daran interessiert, Personen zu kontaktieren, die ein bestimmtes Quellpaket mittels Abschnitt 4.10 abonniert haben. Sie können dazu die E-Mail-Adresse *Paket*`@packages.qa.debian.org` benutzen.

7.4 Sich mit inaktiven und/oder nicht erreichbaren Paketbetreuern beschäftigen

Falls Sie bemerken, dass es einem Paket an Betreuung mangelt, sollten Sie sicherstellen, dass der Betreuer aktiv ist und weiter an seinen Paketen arbeitet. Es ist möglich, dass er nicht mehr aktiv ist, sich aber nicht aus dem System abgemeldet hat. Andererseits ist es auch möglich, dass er nur eine Erinnerung braucht.

Es gibt ein einfaches System (die MIA-Datenbank), in der Informationen über Paketbetreuer aufgezeichnet werden, die als »Missing In Action« (vermisst) gelten. Wenn ein Mitglied der QS-Gruppe einen inaktiven Betreuer kontaktiert oder weitere Informationen über ihn findet, wird dies in der MIA-Datenbank aufgezeichnet. Das System ist unter `/org/qa.debian.org/mia` auf dem Rechner `qa.debian.org` verfügbar und kann mit dem Werkzeug **mia-query** abgefragt werden. Benutzen Sie **mia-query --help**, um zu erfahren, wie Sie Abfragen an die Datenbank richten. Falls Sie der Meinung sind, dass noch keine Informationen über einen inaktiven Betreuer aufgezeichnet wurde oder dass Sie weitere Informationen hinzufügen können, sollten Sie im Allgemeinen wie folgt verfahren.

Im ersten Schritt kontaktieren Sie den Betreuer höflich und warten eine angemessene Zeit auf eine Antwort. Es ist ziemlich schwer zu definieren, was eine angemessene Zeit ist, aber es ist wichtig zu berücksichtigen, dass das wahre Leben manchmal sehr hektisch ist. Eine Möglichkeit damit umzugehen wäre es, nach zwei Wochen eine Erinnerung zu senden.

Falls der Betreuer nicht innerhalb von vier Wochen (einem Monat) antwortet, kann davon ausgegangen werden, dass wahrscheinlich keine Antwort mehr kommt. Falls dies geschieht, sollten Sie weiter nachforschen und versuchen so viele nützliche Informationen wie möglich über den betreffenden Betreuer zu sammeln. Dies beinhaltet:

- die `echelon`-Informationen, die über die debian.org Developers LDAP Search verfügbar sind. Sie geben an, wann der Entwickler zum letzten Mal an eine Debian-Mailingliste geschrieben hat. (Dies umfasst Mails über Uploads, die über die Liste debian-devel-changes@lists.debian.org verteilt wurden.) Denken Sie außerdem daran zu prüfen, ob der Betreuer in der Datenbank als im Urlaub befindlich markiert ist.
- die Zahl der Pakete, für die dieser Betreuer verantwortlich ist und den Zustand dieser Pakete. Hauptsächlich, ob es veröffentlichungskritische Fehler gibt, die seit Jahren offen sind. Ferner wie viele Fehler es im Allgemeinen sind. Ein weiteres wichtiges Teil der Informationen ist, ob für die Pakete NMUs durchgeführt werden und wenn, von wem.
- Gibt es irgendeine Aktivität des Betreuers außerhalb von Debian? Er könnte zum Beispiel aktuell etwas an eine Nicht-Debian-Mailingliste oder an Newsgroups geschrieben haben.

Ein ziemliches Problem sind Pakete, die gesponsort wurden — der Betreuer ist kein offizieller Debian-Entwickler. Die `echelon`-Informationen sind nicht für gesponsorte Leute verfügbar, so dass Sie beispielsweise den Debian-Entwickler finden und kontaktieren müssen, der das Paket tatsächlich hochgeladen hat. Angenommen, das Paket

wurde signiert, dann ist er dennoch für das Paket verantwortlich und weiß wahrscheinlich, was mit der Person geschah, die er sponsorte.

Es ist außerdem erlaubt, eine Anfrage an debian-devel@lists.debian.org zu senden und zu fragen, ob jemand etwas über den Verbleib des vermissten Betreuers weiß. Bitte senden Sie der Person, um die es geht, per Cc: eine Kopie.

Sobald Sie als dies gesammelt haben, können Sie mia@qa.debian.org kontaktieren. Leute mit diesem Alias werden die von Ihnen bereitgestellten Informationen nutzen, um zu entscheiden, wie verfahren wird. Beispielsweise könnten Sie eines oder alle Pakete des Betreuers verwaisen. Falls für ein Paket ein NMU durchgeführt wurde, könnten sie es vorziehen vor dem Verwaisen des Pakets denjenigen zu kontaktieren, von dem der NMU durchgeführt wurde — vielleicht ist diese Person daran interessiert das Paket zu übernehmen.

Eine abschließende Bemerkung: Bitte denken Sie daran, höflich zu bleiben. Alle hier sind Freiwillige und können nicht sämtliche Zeit Debian widmen. Außerdem wissen Sie nichts über die Situation der beteiligten Person. Vielleicht ist sie ernsthaft erkrankt oder sogar verstorben — Sie wissen nicht, wer auf der Empfängerseite ist. Stellen Sie sich vor, wie sich ein Verwandter fühlt, wenn er die E-Mail des Verstorbenen liest und eine sehr unhöfliche, böse oder vorwurfsvolle Nachricht findet!

Andererseits besteht trotz Freiwilligkeit auch eine Verantwortung. So können Sie die Wichtigkeit eines übergeordneten Wohls betonen — falls ein Betreuer keine Zeit oder kein Interesse mehr hat, sollte er gehen und das Paket jemandem mit mehr Zeit geben.

Falls Sie Interesse an der Arbeit im MIA-Team haben, werfen Sie bitte einen Blick in die `README`-Datei in `/org/qa.debian.org/mia` auf `qa.debian.org`, wo die technischen Einzelheiten und MIA-Prozeduren dokumentiert sind und kontaktieren Sie mia@qa.debian.org.

7.5 Zusammenwirken mit zukünftigen Debian-Entwicklern

Debians Erfolg hängt von der Fähigkeit ab, neue und talentierte Entwickler für sich zu gewinnen und zu halten. Wenn Sie ein erfahrener Entwickler sind, wird Ihnen empfohlen, sich am Prozess, neue Entwickler heranzuziehen, zu beteiligen.

7.5.1 Pakete sponsoren

Sponsoring eines Pakets bedeutet, ein Paket für einen Betreuer hochzuladen, der nicht in der Lage ist, dies selbst zu tun. Es ist keine belanglose Angelegenheit, der Sponsor muss die Paketierung überprüfen und dafür sorgen, dass sie auf der hohen Qualitätsstufe ist, die Debian anstrebt.

Debian-Entwickler können Pakete sponsoren. Debian-Betreuer können dies nicht.

Der Prozess, ein Paket zu sponsoren ist:

1. Der Betreuer bereitet ein Quellpaket (`.dsc`) vor und legt es irgendwo online ab (wie auf mentors.debian.net) oder stellt, was noch besser ist, einen Link zu einem öffentlichen VCS-Depot (siehe Abschnitt 4.4.5) bereit, wo das Paket verwaltet wird.

2. Der Sponsor lädt das Paket herunter (oder checkt es aus).

3. Der Sponsor prüft das Quellpaket. Falls er Probleme entdeckt, informiert er den Betreuer und bittet ihn, eine korrigierte Version bereitzustellen (der Prozess beginnt von vorn mit dem ersten Schritt).

4. Der Sponsor konnte kein verbliebenes Problem finden. Er erstellt das Paket, signiert es und lädt es zu Debian hoch.

Bevor Sie die Einzelheiten erforschen, wie ein Paket gesponsort wird, sollten Sie sich selbst fragen, ob das Hinzufügen des angebotenen Pakets einen Gewinn für Debian bedeuted.

Es gibt keine einfache Regel, um diese Frage zu beantworten, sie kann von vielen Faktoren abhängen: Ist die Code-Basis der Originalautoren ausgereift und nicht voller Sicherheitslücken? Gibt es vorher existierende Pakete, die die gleiche Aufgabe erledigen können und wie sind diese im Vergleich zu diesem neuen Paket? Gab es für dieses neue Paket eine Nachfrage durch Anwender und wie groß ist die Benutzerbasis? Wie aktiv sind die Entwickler des Originals?

Sie sollten außerdem sicherstellen, dass der zukünftige Betreuer ein guter Betreuer sein wird. Hat er bereits etwas Erfahrung mit anderen Paketen? Falls ja, leistet er bei ihnen gute Arbeit (überprüfen Sie einige Fehler)? Ist er vertraut mit dem Paket und dessen Programmiersprache? Verfügt er über die für dieses Paket nötigen Fähigkeiten? Falls nicht, ist er in der Lage sie zu erlernen?

Es ist außerdem eine gute Idee zu wissen, wie er Debian gegenübersteht: Stimmt er der Debian-Philosophie zu und beabsichtigt er Debian beizutreten? Angesichts dessen, wie einfach es ist, ein Debian-Betreuer zu werden, möchten Sie vielleicht nur Leute sponsoren, die planen beizutreten. Auf diese Art wissen Sie von Beginn an, dass Sie nicht auf unbestimmte Zeit als Sponsor agieren wollen.

7.5.1.1 Ein neues Paket sponsoren

Neue Betreuer haben gewöhnlich bestimmte Schwierigkeiten bei der Erstellung von Debian-Paketen — dies ist ziemlich verständlich. Das ist der Grund, weshalb das Sponsoring eines brandneuen Pakets in Debian, eine sorgfältige Überprüfung notwendig macht. Manchmal sind mehrere Wiederholungen nötig, bis das Paket gut genug ist, um zu Debian hochgeladen zu werden. Daher impliziert ein Sponsor zu sein, dass man ein Mentor ist.

Sponsoren Sie ein Paket nie, ohne es zu überprüfen. Die Überprüfung eines neuen Pakets, die von den Ftpmasters vorgenommen wird, stellt nur sicher, dass die Software wirklich frei ist. Natürlich kommt es vor, dass sie über Paketierungsprobleme stolpern, aber sie sollten es wirklich nicht. Es ist Ihre Aufgabe dafür zu sorgen, dass das hochgeladene Paket mit den Richtlinien Debians für freie Software übereinstimmt und von guter Qualität ist.

Das Erstellen des Pakets und Prüfen der Software ist Teil der Überprüfung, aber es reicht nicht aus. Der Rest dieses Abschnitts enthält eine unvollständige Liste von Punkten, die Sie in Ihrer Überprüfung testen sollten. [1]

- Prüfen Sie, ob der bereitgestellte Original-Tarball derselbe ist, wie der, den der Originalautor verteilt (wenn die Quellen neu für Debian gepackt wurden, erstellen Sie selbst den geänderten Tarball).
- Führen Sie **lintian** aus (siehe Abschnitt A.2.1). Es wird viele häufige Probleme erwischen. Achten Sie darauf zu überprüfen, dass jegliche Einstellung, durch die der Betreuer **lintian** außer Kraft setzt, vollständig gerechtfertigt ist.
- Führen Sie **licensecheck** aus (Teil von Abschnitt A.6.1) und überprüfen Sie, ob `debian/copyright` korrekt und komplett zu sein scheint. Suchen Sie nach Lizenzproblemen (wie Dateien mit »All rights reserved«-Kopfzeilen oder nicht DFSG-konformen Lizenzen). Bei dieser Aufgabe ist **grep -ri** Ihr Freund.
- Erstellen Sie das Paket mit **pbuilder** (oder einem ähnlichen Werkzeug, siehe Abschnitt A.4.3), um sicherzustellen, dass die Build-Abhängigkeiten vollständig sind.
- Lesen Sie `debian/control` Korrektur: Folgt es den optimalen Vorgehensweisen (siehe Abschnitt 6.2)? Sind die Abhängigkeiten vollständig?
- Lesen Sie `debian/rules` Korrektur: Folgt es den optimalen Vorgehensweisen (siehe Abschnitt 6.1)? Können Sie mögliche Verbesserungen sehen?
- Lesen Sie die Betreuerskripte (`preinst`, `postinst`, `prerm`, `postrm`, `config`) Korrektur: Werden `preinst`/`postrm` funktionieren, wenn die Abhängigkeiten nicht installiert sind? Sind alle Skripte idempotent (d.h. können Sie sie mehrmals ohne Konsequenzen ausführen)?
- Überprüfen Sie jede Änderung an Originaldateien (entweder in `.diff.gz`, in `debian/patches/` oder direkt in den Tarball `debian` für Binärdateien eingebettet). Sind sie gerechtfertigt? Sind sie ordentlich dokumentiert (mit DEP-3 für Patche)?
- Fragen Sie sich für jede Datei, warum diese Datei dort ist und ob dies der richtige Weg ist das gewünschte Ergebnis zu erzielen. Folgt der Betreuer den optimalen Vorgehensweisen beim Paketieren (siehe Kapitel 6)?
- erstellen Sie die Pakete, installieren Sie sie und probieren Sie die Software aus. Stellen Sie sicher, dass Sie die Pakete entfernen und vollständig beseitigen können. Eventuell testen Sie sie mit **piuparts**.

Falls die Kontrolle kein Problem offenbarte, können Sie das Paket erstellen und zu Debian hochladen. Denken Sie daran, dass , obwohl Sie nicht der Betreuer sind, der Sponsor immer noch für das vernatwortlich ist, was er zu Debian hochlädt. Daher sei Ihnen geraten, das Paket durch das Abschnitt 4.10 im Auge zu behalten.

Beachten Sie, dass Sie das Quellpaket nicht verändern müssen, um Ihren Namen in die Datei `changelog` oder `control` einzutragen. Das Feld `Maintainer` der Dateien `control` und `changelog` sollte die Person aufführen, die das Paket erstellte, d.h. den Gesponsorten. Auf diese Art wird er daher alle Mail des BTS erhalten.

Stattdessen sollten Sie **dpkg-buildpackage** anweisen, Ihren Schlüssel für die Signatur zu benutzen. Sie erreichen dies mit der Option `-k`:

[1] Sie können weitere Überprüfungen im Wiki finden, wo mehrere Entwickler ihre eigenen Sponsorschaft-Prüflisten miteinander teilen.

```
dpkg-buildpackage -kSCHLÜSSELNUMMER
```

Falls Sie **debuild** und **debsign** benutzen, können sie das sogar permanent in `~/.devscripts` konfigurieren:

```
DEBSIGN_KEYID=SCHLÜSSELNUMMER
```

7.5.1.2 Eine Aktualisierung eines existierenden Pakets sponsoren

Sie werden normalerweise davon ausgehen, dass das Paket bereits eine vollständige Überprüfung durchlief. Daher werden Sie, anstatt dies wieder zu tun, die Unterschiede zwischen der akutellen und der neu vom Betreuer vorbereiteten Version sorgsam analysieren. Falls Sie die anfängliche Überprüfung nicht selbst durchgeführt haben, können Sie immer noch einen genaueren Blick darauf werfen, nur für den Fall, dass der erste Prüfer nachlässig war.

Um in der Lage zu sein, die Unterschiede zu untersuchen, benötigen Sie beide Versionen. Laden Sie die aktuelle Version des Quellpakets herunter (mit **apt-get source**) und erstellen Sie es (oder laden Sie die aktuellen Binärpakete mit **aptitude download** herunter). Laden Sie das Quellpaket zum Sponsorn (üblicherweise mit **dget**).

Lesen Sie den Änderungsprotokolleintrag, er sollte mitteilen, was Sie während der Überprüfung erwartet. Das Hauptwerkzeug, das Sie benutzen werden, ist **debdiff** (bereitgestellt vom Paket `devscripts`). Sie können es mit zwei Quellpaketen (`.dsc`-Dateien), zwei Binärpaketen oder zwei `.changes`-Dateien (dann wird es alle Binärpakete vergleichen, die in `.changes` aufgeführt sind) ausführen.

Falls Sie die Quellpakete vergleichen (ausschließlich der Originaldateien im Fall einer neuen Originalversion, zum Beispiel durch Filtern der Ausgabe von **debdiff** mit **filterdiff -i '*/debian/*'**), müssen Sie alle Änderungen verstehen und sie sollten ordentlich im Debian-Änderungsprotokoll dokumentiert sein.

Falls alles in Ordnung ist, erstellen Sie das Paket und vergleichen Sie die Binärpakete, um zu prüfen, ob die Änderungen am Quellpaket keine unerwarteten Folgen haben (wie einige versehentlich entfallenen Dateien, fehlende Abhängigkeiten etc.

Sie möchten möglicherweise das Package-Tracking-System austesten (siehe Abschnitt 4.10), um zu überprüfen, ob der Betreuer nichts Wichtiges versäumt hat. Möglicherweise gibt es Übersetzungsaktualisierungen, die im BTS liegen und hätten integriert werden können. Möglicherweise wurde ein NMU des Pakets durchgeführt und der Betreuer vergaß, die Änderungen des NMUs in sein Paket einzubauen. Vielleicht ist ein veröffentlichungskritischer Fehler unbehandelt geblieben und dies blockiert die Migration nach `testing`. Was auch immer. Falls Sie etwas finden, was (besser) erledigt werden könnte, ist es an der Zeit ihm dies mitzuteilen, so dass er es das nächste Mal besser machen kann und dass er seine Verantwortlichkeiten besser versteht.

Falls Sie kein bedeutendes Problem gefunden haben, laden Sie die neue Version hoch. Andernfalls bitten Sie den Betreuer, Ihnen eine reparierte Version zur Verfügung zu stellen.

7.5.2 Neue Entwickler befürworten

Lesen Sie die Seite Einen zukünftigen Entwickler befürworten auf der Debian-Website.

7.5.3 Handhabung von Bewerbungen neuer Betreuer

Lesen Sie bitte die Checkliste für Bewerbungsleiter auf der Debian-Website.

Kapitel 8

Internationalisierung und Übersetzungen

Debian unterstützt eine immer größer werdende Zahl natürlicher Sprachen. Selbst wenn Englisch Ihre Muttersprache ist und Sie keine andere Sprache sprechen, gehört es zu Ihren Pflichten als Paketbetreuer, die Probleme der Internationalisierung zu kennen (abgekürzt I18n, weil 18 Buchstaben zwischen »i« und »n« im englischen Wort »internationalization« stehen). Daher sollten Sie sogar, wenn Sie mit rein englischen Programmen klarkommen, das meiste in diesem Kapitel lesen.

Gemäß der Einführung in I18n von Tomohiro KUBOTA bedeutet I18n (Internationalisierung) eine Veränderung von Software oder damit verbundenen Technologien, so dass eine Software potentiell mehrere Sprachen, Gewohnheiten und so weiter in der Welt handhaben kann, während L10n (Lokalisierung) die Implementierung einer speziellen Sprache für eine bereits internationalisierte Software bedeutet.

L10n und I18n sind miteinander verbunden, aber die Schwierigkeiten bezogen auf jeweils eines davon sind sehr unterschiedlich. Es ist nicht wirklich schwer, einem Programm das Wechseln der Sprache zu erlauben, in dem Texte basierend auf den Benutzereinstellungen angezeigt werden, aber es verschlingt viel Zeit, diese Nachrichten tatsächlich zu übersetzen. Andererseits ist es trivial, die Zeichencodierung einzustellen, aber den Code so anzupassen, dass mehrere Zeichencodierungen benutzt werden können, ist ein wirklich großes Problem.

Abgesehen von den I18n-Problemen, für die keine allgemeine Anleitung gegeben werden kann, gibt es tatsächlich keine Infrastruktur für L10n innerhalb Debian, die mit der Buildd-Mechanisus für die Portierung vergleichbar ist. Daher muss die meiste Arbeit manuell erledigt werden.

8.1 Wie Übersetzungen in Debian gehandhabt werden

Die Handhabung der Übersetzung von Texten, die in Paketen enthalten sind, ist immer noch eine manuelle Aufgabe und der Prozess hängt von der Art des Textes ab, den Sie übersetzt sehen wollen.

Für Programmausgaben wird meistens die Gettext-Infrastruktur benutzt. Meistens wird die Übersetzung außerhalb von Debian in Projekten wie dem Free Translation Project, dem Gnome translation Project oder der KDE Localization behandelt. Die einzige zentrale Ressource innerhalb Debian ist die Zentrale Übersetzungsstatistik von Debian, wo Sie einige Statistiken über die Übersetzungsdateien finden können, die in den tatsächlichen Paketen gefunden wurden, aber keine echte Infrastruktur, um den Übersetzungsprozess zu erleichtern.

Es wurde vor langer Zeit ein Versuch gestartet, die Paketbeschreibungen zu übersetzen, auch wenn sehr wenig Unterstützung von Werkzeugen angeboten wird, um sie zu benutzen (d.h. nur APT kann sie nutzen, wenn es korrekt konfiguriert ist). Betreuer müssen nichts besonderes tun, um übersetzte Paketbeschreibungen zu unterstützen. Übersetzer sollten das Debian Description Translation Project (DDTP) benutzen.

Für `debconf`-Schablonen sollten Paketbetreuer das Paket `po-debconf` benutzen, um die Arbeit der Übersetzer zu erleichtern, die das DDTP für ihre Arbeit benutzen könnten. (Das französische und das brasilianische Team nutzen dies nicht.). Einige Statistiken können sowohl auf der DDTP-Site (über das, was tatsächlich übersetzt ist) als auch in der Zentralen Übersetzungsstatistik von Debian (über das, was in die Pakete eingegliedert ist) gefunden werden.

Für Webseiten hat jedes L10n-Team Zugriff auf das passende VCS und die Statistiken sind auf der Site bei der zentralen Übersetzungsstatistik von Debian verfügbar.

Für allgemeine Dokumentation über Debian entspricht der Prozess mehr oder weniger dem der Webseiten (für Übersetzer, die Zugriff auf das VCS haben), aber es gibt dort keine Statistikseiten.

Für paketspezifische Dokumentation (Handbuchseiten, Info-Dokumente, andere Formate) bleibt beinahe alles zu erledigen.

Insbesondere das KDE-Projekt handhabt die Übersetzung seiner Dokumentation auf die gleiche Art, wie die der Programmausgaben.

Es gibt einen Versuch, Debian-spezifische Handbuchseiten innerhalb eines speziellen VCS-Depots zu handhaben.

8.2 I18N & L10N FAQ für Paketbetreuer

Dies ist eine Liste von Problemen, denen Betreuer betreffend I18n und L10n gegenüberstehen. Behalten Sie während Sie dies lesen im Hinterkopf, dass es über diese Punkte innerhalb Debian keine Einigkeit gibt und dies nur ein Ratschlag ist. Falls Sie für die vorliegenden Probleme bessere Ideen haben oder mit einigen Punkten nicht einverstanden sind, tun Sie sich keinen Zwang an und geben Sie Ihre Rückmeldung, so dass dieses Dokument verbessert werden kann.

8.2.1 Wie ein vorliegender Text übersetzt wird

Um Paketbeschreibungen oder `debconf`-Schablonen zu übersetzen, müssen Sie nichts tun. Die DDTP-Infrastruktur wird das zu übersetzende Material zu den Freiwilligen befördern ohne dass Sie eingreifen müssen.

Für jegliches andere Material (Gettext-Dateien, Handbuchseiten oder andere Dokumentation) ist die beste Lösung, den Text irgendwo ins Internet zu stellen und auf Debian-I18n um eine Übersetzung in verschiedene Sprachen zu ersuchen. Die Mitglieder einiger Übersetzer-Teams haben diese Liste abonniert und werden für die Übersetzung und den Korrekturprozess sorgen. Sobald Sie das getan haben, können Sie Ihr übersetztes Dokument der Mailbox entnehmen.

8.2.2 Wie eine vorliegende Übersetzung überprüft wird

Von Zeit zu Zeit übersetzen Einzelpersonen einige Texte in Ihrem Paket und bitten Sie, die Übersetzung in das Paket aufzunehmen. Dies kann problematisch werden, falls Sie die vorliegende Sprache nicht fließend sprechen. Es ist eine gute Idee, das Dokument an die entsprechende L10n-Mailingliste zu senden und um eine Überprüfung zu bitten. Sobald dies erledigt ist, sollten Sie von der Qualität der Übersetzug überzeugt sein und sich sicherer fühlen sie in Ihr Paket einzubinden.

8.2.3 Wie eine vorliegende Übersetzung aktualisiert wird

Falls Sie einige Übersetzungen eines vorliegendens Textes herumliegen haben, sollten Sie jedesmal, wenn Sie das Original aktualisieren, den letzten Übersetzer bitten die Übersetzung mit Ihren neuen Änderungen zu aktualisieren. Behalten Sie im Hinterkopf, dass diese Aufgabe Zeit beansprucht; mindestens eine Woche um die Aktualisierung zu überprufen und so.

Falls der Übersetzer nicht reagiert, könnten Sie auf der entsprechenden L10n-Mailingliste um Hilfe ersuchen. Falls alles scheitert, vergessen Sie nicht, eine Warnung im übersetzten Dokument zu hinterlassen, die angibt, dass die Übersetzung veraltet ist und der Leser sich, wenn möglich, auf das Originaldokument beziehen sollte.

Vermeiden Sie es, eine Übersetzung vollständig zu entfernen, weil sie veraltet ist. Alte Dokumentation ist für Leute, die kein Englisch reden, oft besser als gar keine Dokumentation

8.2.4 Wie Fehlerberichte gehandhabt werden, die eine Übersetzung betreffen

Möglicherweise ist die beste Lösung, den Fehler als zu den Originalautoren weitergeleitet zu kennzeichnen und ihn sowohl an den letzten Übersetzer als auch an sein Team zu senden (unter Benutzung der entsprechenden debian-l10n-XXX-Mailingliste).

8.3 I18n- & L10n-FAQ für Übersetzer

Behalten Sie während Sie dies lesen im Hinterkopf, dass es über diese Punkte innerhalb Debian keine Einigkeit gibt und dass Sie auf jeden Fall mit Ihrem Team und dem Paketbetreuer zusammenarbeiten sollten.

8.3.1 Wie bei Übersetzungsbemühungen geholfen wird

Wählen Sie aus, was Sie übersetzen möchten und stellen Sie sicher, dass nicht bereits jemand daran arbeitet (benutzen Sie Ihre debian-l10n-XXX-Mailingliste), übersetzen Sie es, lassen Sie es durch andere mit dieser Muttersprache auf Ihrer L10n-Mailingliste überprüfen und stellen Sie es dem Paketbetreuer zur Verfügung (siehe nächsten Punkt).

8.3.2 Wie eine Übersetzung zur Eingliederung in ein Paket bereitgestellt wird

Stellen Sie sicher, dass Ihre Übersetzung korrekt ist (bitten Sie auf Ihrer L10n-Mailingliste um eine Überprüfung), bevor Sie sie zur Eingliederung bereitgestellen. Es wird für jeden eine Zeitersparnis sein und das Chaos vermeiden, das daraus resultiert, dass Sie mehrere Versionen des gleichen Dokuments in Fehlerberichten haben.

Die beste Lösung ist es, einen regulären Fehlerbericht gegen das Paket einzureichen, der die Übersetzung enthält. Stellen Sie sicher, dass Sie die Kennzeichnung »PATCH« und keinen Schweregrad höher als »wishlist« verwenden, da das Fehlen einer Übersetzung ein Programm niemals an der Ausführung hindert.

8.4 Beste aktuelle Vorgehensweise bezüglich L10n

- Bearbeiten Sie als Betreuer niemals die Übersetzungen in irgendeiner Weise (auch um das Aussehen neu zu formatieren) ohne mit der entsprechenden L10n-Mailingliste zu sprechen. Sie riskieren zum Beispiel, die Zeichencodierung der Datei zu zerstören, wenn Sie dies tun. Außerdem könnte das, was Sie als Fehler betrachen, in der vorliegenden Sprache richtig sein (oder sogar nötig).
- Falls Sie als Übersetzer einen Fehler im Originaltext finden, stellen Sie sicher, dass er gemeldet wird. Übersetzer sind oft die aufmerksamsten Leser eines vorliegenden Textes und falls Sie die gefundenen Fehler nicht melden, wird es niemand tun.
- Denken Sie auf jeden Fall daran, dass der Hauptstreitpunkt mit L10n darin besteht, dass mehrere Leute für eine Zusammenarbeit nötig sind und dass es sehr leicht ist, einen Flame-War über kleine Probleme aufgrund von Missverständnissen heraufzubeschwören. Falls Sie daher Probleme mit einem Gesprächspartner haben, fragen Sie auf der entsprechenden L10n-Mailingliste, Debian-I18n oder sogar Debian-Devel nach Hilfe (aber Vorsicht, L10n-Diskussionen führen auf dieser Liste oft zu Flame-Wars :)).
- Kooperation kann auf jeden Fall nur mit **gegenseitigem Respekt** erreicht werden.

Anhang A

Überblick über die Werkzeuge der Debian-Betreuer

Dieser Abschnitt enthält eine grobe Übersicht über die Werkzeuge, die Betreuern zur Verfügung stehen. Das Folgende ist beileibe nicht vollständig oder maßgeblich, sondern nur eine Anleitung für einige der beliebstesten Werkzeuge.

Debian-Betreuerwerkzeuge sind dazu gedacht, Entwicklern zu helfen und Zeit für wirklich kritische Aufgaben einzuräumen. Wie schon Larry Wall sagte, gibt es mehr als einen Weg, um etwas zu erledigen.

Einige Leute bevorzugen die Benutzung von hochrangigen Paketverwaltungswerkzeugen, andere nicht. Debian ist bei diesem Thema agnostisch; jedes Werkzeug, das seine Aufgabe erfüllt, ist gut. Daher ist dieser Abschnitt nicht dazu gedacht, jemandem vorzuschreiben, welche Werkzeuge er benutzen oder wie er mit seinen Pflichen als Betreuer umgehen soll. Er ist auch nicht dazu gedacht, ein besonderes Werkzeug zu befürworten, um ein konkurrierendes auszuschließen.

Die meisten Beschreibungen dieser Pakete entstammen selbst den tatsächlichen Paketbeschreibungen. Weitere Informationen können in der Paketbeschreibung selbst gefunden werden. Sie können außerdem mit dem Befehl **apt-cache show *`Paketname`*** zusätzliche Informationen lesen.

A.1 Kernwerkzeuge

Die folgenden Werkzeuge werden größtenteils von jedem Betreuer benötigt.

A.1.1 `dpkg-dev`

`dpkg-dev` enthält die Werkzeuge (einschließlich **dpkg-source**), die benötigt werden, um Debian-Pakete zu entpacken, zu erstellen und hochzuladen. Diese Hilfswerkzeuge enthalten die untergeordneten Funktionalitäten, die zum Erstellen und Manipulieren von Paketen benötigt werden; als solches sind sie für jeden Debian-Betreuer erforderlich.

A.1.2 `debconf`

`debconf` stellt eine einheitliche Schnittstelle zur Verfügung, um Pakete interaktiv zu konfigurieren, Es ist unabhängig von der Schnittstelle des Anwenders, erlaubt Endanwendern Pakete mit einer reinen Textschnittstelle, einer HTML-Schnittstelle oder einer Dialogschnittstelle zu konfigurieren. Neue Schnittstellen können als Module hinzugefügt werden.

Sie können Dokumentation für dieses Paket im Paket `debconf-doc` finden.

Viele sind der Ansicht, dieses System sollte für alle Pakete verwandt werden, die eine interaktive Konfiguration erfordern; siehe Abschnitt 6.5. Derzeit wird `debconf` noch nicht von den Debian Richtlinien benötigt, aber das kann sich in der Zukunft ändern.

A.1.3 `fakeroot`

`fakeroot` simuliert Root-Rechte. Dies ermöglicht Ihnen, Pakete zu erstellen ohne Root zu sein (Pakete möchten üblicherweise Dateien mit Root-Besitzrechten installieren). Falls Sie `fakeroot` installiert haben, können Sie Pakete als normaler Anwender erstellen: `dpkg-buildpackage -rfakeroot`.

A.2 Lint-Werkzeuge für Pakete

Gemäß dem Free On-line Dictionary of Computing (FOLDOC) ist »lint« ein Unix-Prozessor für die Sprache C, der gründlichere Prüfungen des Codes mitbringt als übliche C-Kompiler. Lint-Werkzeuge für Pakete helfen Paketbetreuern automatisch häufige Probleme und Richtlinienverletzungen in ihren Paketen zu finden.

A.2.1 `lintian`

`lintian` zerlegt Debian-Pakete und gibt Informationen über Fehler und Richtlinien-Verletzungen aus. Es enthält automatisierte Prüfungen für viele Gesichtspunkte der Debian-Richtlinien, als auch einige Prüfungen für häufige Fehler.

Sie sollten regelmäßig das neuste `lintian` aus `unstable` besorgen und all Ihre Pakete überprüfen. Beachten Sie, dass die Option `-i` detaillierte Erklärungen liefert, was jeder Fehler oder jede Warnung bedeutet, was die Grundlage in der Richtlinie ist und wie das Problem üblicherweise behoben werden kann.

Es sei für weitere Informationen darüber, wie und wann Lintian benutzt wird, auf Abschnitt 5.3 verwiesen.

Sie können außerdem eine Zusammenfassung aller Probleme, die Lintian in all Ihren Paketen meldet unter `http://lintian.debian.org/` ansehen. Diese Berichte enthalten die letzte Ausgabe von **lintian** für die ganze Entwicklungsdistribution (`unstable`).

A.2.2 debdiff

debdiff (aus dem Paket `devscripts`, Abschnitt A.6.1) vergleicht die Dateilisten und »control«-Dateien zweier Pakete. Es ist ein einfacher Rückfalltest, der Ihnen hilft festzustellen, ob sich die Anzahl der Binärpakete seit dem letzten Upload verändert hat oder ob sich etwas in der »control«-Datei geändert hat. Natürlich werden einige Änderungen, die es meldet, in Ordnung sein, aber es kann Ihnen helfen verschiedene Unfälle zu verhüten.

Sie können es für ein Paar binärer Pakete ausführen:

```
debdiff package_1-1_arch.deb package_2-1_arch.deb
```

oder sogar für ein Paar aus »changes«-Dateien:

```
debdiff package_1-1_arch.changes package_2-1_arch.changes
```

Um weitere Informationen zu erhalten, lesen Sie debdiff(1).

A.3 Helper-Skripte für `debian/rules`

Paketerstellungswerkzeuge erleichtern das Verfassen von `debian/rules`-Dateien. Lesen Sie Abschnitt 6.1.1, um weitere Informationen darüber zu erhalten, warum dies erwünscht und jenes unerwünscht sein könnte.

A.3.1 `debhelper`

`debhelper` ist eine Programmsammlung, die in `debian/rules` benutzt werden kann, um häufige Aufgaben zu automatisieren, die sich auf das Erstellen binärer Debian-Pakete beziehen. `debhelper` enthält Programme, um verschiedene Dateien in Ihre Pakete zu installieren, Dateien zu komprimieren, Dateirechte zu korrigieren und Ihr Paket in das Debian-Menüsystem zu integrieren.

Anders als bei einigen Herangehensweisen ist `debhelper` in mehrere kleine einfache Befehle unterteilt, die auf eine durchgängige Art zusammenarbeiten. Als solches erlaubt es eine detailliertere Steuerung, als andere Werkzeuge für »debian/rules«.

Es gibt eine zu große Zahl kleiner Erweiterungspakete für `debhelper`, die zu kurzlebig sind, um sie zu dokumentieren. Sie können die Liste der meisten von Ihnen ansehen, indem Sie `apt-cache search ^dh-` aufrufen.

A.3.2 `dh-make`

Das Paket `dh-make` enthält ein Programm gleichen Namens, das ein Gerüst von Dateien erstellt, die nötig sind, um Debian-Pakete aus einem Quellcodeverzeichnisbaum zu erstellen. Wie der Name schon nahelegt, ist **dh_make** eine Neufassung von `debmake` dessen Schablonendateien **dh_***-Programme von `debhelper` benutzen.

Während die von **dh_make** generierten »rules«-Dateien im Allgemeinen eine ausreichende Basis für ein funktionierendes Paket bilden, gibt es immer noch die grundlegenden Arbeiten: Die Last für die Feinabstimmung und das Paket funktional und richtlinienkonform zu machen, liegt immer noch beim Betreuer.

A.3.3 equivs

equivs ist ein weiteres Paket für die Paketerstellung. Es wird oft für den lokalen Gebrauch vorgeschlagen, falls Sie einfach ein Paket erstellen müssen, um Abhängigkeiten zu erfüllen. Es wird manchmal auch benutzt, um »Meta-Pakete« zu erstellen. Dabei handelt es sich um Pakete, deren einziger Zweck darin besteht, von anderen Paketen abzuhängen.

A.4 Paket-Builder

Die folgenden Pakete helfen beim Prozess der Paketerstellung und führen im Allgemeinen **dpkg-buildpackage** aus, um unterstützende Aufgaben zu behandeln.

A.4.1 cvs-buildpackage

cvs-buildpackage stellt die Fähigkeit bereit, wichtige Debian-Quellpakete in ein CVS-Depot einzuspeisen, ein Debian-Paket aus dem CVS-Depot zu erstellen und bei der Integration von Änderungen der Originalautoren in das CVS-Depot zu helfen.

Diese Hilfswerkzeuge bieten eine Infrastruktur, um Debian-Betreuern den Gebrauch von CVS zu erleichtern. Dies ermöglicht getrennte Zweige von Paketen für die Distributionen stable, unstable und möglicherweise experimental vorzuhalten, zusammen mit den anderen Vorteilen eines Versionsverwaltungssystems.

A.4.2 debootstrap

Das Paket und Skript debootstrap ermöglicht Ihnen das Urladen eines Debian-Basissystems in irgendeinen Teil Ihres Dateisystems. Mit Basissystem ist ein Minimum an installierten Paketen gemeint, die nötig sind, um des Rest des Systems zu betreiben und zu installieren.

Ein solches System zu haben, kann in vielerlei Hinsicht nützlich sein. Sie können zum Beispiel mit **chroot** in das System gehen und wenn Sie wollen, Ihre Build-Abhängigkeiten testen. Oder Sie können testen, wie sich Ihr Paket verhält, wenn es in ein nacktes Basissystem installiert wird. Chroot-Builder benutzen dieses Paket; siehe Folgendes.

A.4.3 pbuilder

pbuilder konstruiert ein Chroot-System und erstellt ein Paket innerhalb der Chroot-Umgebung. Es ist sehr nützlich, um zu prüfen, ob die Build-Abhängigkeiten des Pakets korrekt sind und um sicher zu sein, dass keine unnötigen oder falschen Build-Abhängigkeiten in dem resultierenden Paket existieren.

Ein verwandtes Paket ist pbuilder-uml, das sogar noch weiter geht, indem es den Build innerhalb einer User-Mode-Linux-Umgebung durchführt.

A.4.4 sbuild

sbuild ist ein weiterer automatisierter Builder. Er kann auch Chroot-Umgebungen benutzen. Er kann eigenständig benutzt werden oder als Teil einer verteilten Build-Umgebung über ein Netzwerk. Als letzteres ist er Teil des Systems, das Portierer benutzen, um Binärpakete für all die verfügbaren Architekturen zu erstellen. Weitere Informationen finden Sie unter Abschnitt 5.10.3.3 und das System können Sie unter http://buildd.debian.org/ in Aktion sehen.

A.5 Programme zum Hochladen von Paketen

Die folgenden Pakete helfen den Prozess, Pakete in das offizielle Archiv hochzuladen, zu automatisieren oder zu vereinfachen.

A.5.1 dupload

dupload ist ein Paket und ein Skript, um Debian-Pakete automatisch in das Debian-Archiv hochzuladen, den Upload zu protokollieren und Mails über den Upload eines Pakets zu versenden. Sie können es für neue Upload-Orte und -Methoden konfigurieren.

A.5.2 dput

Das Paket und Skript `dput` tut das gleiche wie `dupload`, aber auf eine andere Art. Es hat einige Funktionalitäten mehr als `dupload`, wie die Fähigkeit GnuPG-Signaturen und Prüfsummen vor dem Upload zu überprüfen und die Möglichkeit nach dem Upload **dinstall** im Leerlaufmodus auszuführen.

A.5.3 dcut

Das Skript **dcut** (Teil des Pakets `dput`, Abschnitt A.5.2) hilft beim Entfernen von Dateien aus dem FTP-Upload-Verzeichnis.

A.6 Verwaltungsautomatisierung

Die folgenden Werkzeuge helfen verschiedene Verwaltungsaufgaben vomHinzufügen von Änderungsprotokolleinträgen oder Signaturzeilen bis zum Nachschlagen von Fehlern in Emacs zum Gebrauch vom neusten und offiziellen `config.sub` zu automatisieren.

A.6.1 devscripts

`devscripts` ist ein Paket, das Wrapper und Werkzeuge enthält, die die sehr hilfreich für die Verwaltung von Debian-Paketen sind. Beispielskripte beinhalten **debchange** und **dch**, die Ihre `debian/changelog`-Datei von der Befehlszeile manipulieren und **debuild**, das ein Wrapper um **dpkg-buildpackage** ist. Außerdem ist das Hilfswerkzeug **bts** sehr hilfreich, um den Status von Fehlerberichten auf der Befehlszeile zu aktualisieren. **uscan** kann benutzt werden, um neue Versionen Ihres Pakets von den Originalautoren zu beobachten. **debrsign** kann benutzt werden, um ein Paket vor dem Upload aus der Ferne zu signieren, was angenehm ist, wenn der Rechner, auf dem Sie das Paket erstellen, sich von dem unterscheidet, auf dem die GPG-Schlüssel liegen.

Eine vollständige Liste der verfügbaren Skripte finden Sie auf der Handbuchseite devscripts(1).

A.6.2 autotools-dev

`autotools-dev` enthält optimale Vorgehensweisen für Leute, die Pakete betreuen, die **autoconf** und/oder **automake** benutzen. Außerdem anthält es vorschriftsmäßige `config.sub`- und `config.guess`-Dateien, von denen bekannt ist, dass sie auf allen Debian-Portierungen funktionieren.

A.6.3 dpkg-repack

dpkg-repack erstellt eine Debian-Paketdatei aus einem Paket, das bereits installiert wurde. Falls irgendwelche Änderungen vorgenommen wurden, während das Pakert entpackt war (es wurden z.B. Dateien in `/etc` verändert), wird das neue Paket die Änderungen erben.

Dieses Hilfswerkzeug kann das Kopieren von Paketen von einem Rechner zu einem anderen, das Neuerstellen von Paketen, die auf Ihrem System installiert wurden, aber nirgendwo mehr verfügbar sind oder das Sichern des derzeitigen Paketstatus vor dem Upgrade vereinfachen.

A.6.4 alien

alien wandelt Binärpakete zwischen verschiedenen Paketformaten, einschließlich Debian, RPM (RedHat), LSB (Linux Standard Base), Solaris und Slackware um.

A.6.5 debsums

debsums überprüft die MD5-Prüfsummen von installierten Paketen. Beachten Sie, dass nicht alle Pakete MD5-Prüfsummen haben, da diese nicht von den Richtlinien vorgeschrieben werden.

A.6.6 dpkg-dev-el

`dpkg-dev-el` ist ein Emacs-Lisp-Paket, das Unterstützung beim Bearbeiten von Dateien im `debian`-Verzeichnis Ihres Pakets bietet. Es gibt dort zum Beispiel praktische Funktionen, um die aktuellen Fehler eines Programm aufzulisten und um den letzten Eintrag in einer `debian/changelog`-Datei zu beenden.

A.6.7 dpkg-depcheck

dpkg-depcheck (aus dem Paket `devscripts`, Abschnitt A.6.1) führt einen Befehl unter **strace** aus, um festzustellen, welche Pakete vom angegebenen Befehl benutzt werden.

Für Debian-Pakete ist dies nützlich, wenn Sie eine `Build-Depends`-Zeile für Ihr neues Paket verfassen müssen: den Build-Prozess durch **dpkg-depcheck** auszuführen wird Sie mit einer guten ersten Annäherung von Build-Abhängigkeiten versorgen. Zum Beispiel:

```
dpkg-depcheck -b debian/rules build
```

dpkg-depcheck kann außerdem benutzt werden um Laufzeitabhängigkeiten zu prüfen, insbesondere, wenn Ihr Paket exec(2) benutzt, um andere Programme auszuführen.

Weitere Informationen finden Sie unter dpkg-depcheck(1).

A.7 Portierungswerkzeuge

Die folgenden Werkzeuge sind hilfreich für Portierer und Kompilierung für andere Plattformen.

A.7.1 `quinn-diff`

`quinn-diff` wird benutzt, um die Unterschiede zwischen zwei Architekturen zu finden. Es könnte zum Beispiel aufzeigen, welche Pakete für die Architektur *`Y`* basierend auf Architektur *`X`* portiert werden müssen.

A.7.2 `dpkg-cross`

`dpkg-cross` ist ein Werkzeug, um Bibliotheken und Header zum Kompilieren auf anderen Plattformen auf eine Art zu installieren, die `dpkg` ähnlich ist. Weiterhin wird die Funktionalität von **dpkg-buildpackage** und **dpkg-shlibdeps** durch die Unterstützung vom Kompilieren auf anderen Plattformen verbessert.

A.8 Dokumentation und Information

Die folgenden Pakete stellen Informationen für Betreuer zur Verfügung oder helfen bei der Erstellung von Dokumentation.

A.8.1 `docbook-xml`

`docbook-xml` stellt die DocBook-XML-DTDen bereit, die häufig für Debian-Dokumentation benutzt werden (genauso wie die ältere Debiandoc-SGML-DTD). Dieses Handbuch wurde zum Beispiel in Docbook-XML verfasst.

Das Paket `docbook-xsl` stellt die XSL-Dateien zum Erstellen und Gestalten der Quelle in verschiedenen Ausgabeformaten bereit. Sie benötigen ein XSLT-Bearbeitungsprogramm wie `xsltproc`, um die XSL-Stylesheets zu verwenden. Dokumentation für die Stylesheets kann in den verschiedenen `docbook-xsl-doc-*`-Paketen gefunden werden.

Um PDF aus FO zu erstellen, benötigen Sie ein FO-Bearbeitungsprogramm wie `xmlroff` oder `fop`. Ein weiteres Werkzeug, um PDF aus DocBook-XML zu generieren ist `dblatex`.

A.8.2 `debiandoc-sgml`

`debiandoc-sgml` stellt die DebianDoc-SGML-DTD bereit, die normalerweise für Debian-Dokumentation benutzt aber nun missbilligt wird (stattdessen sollte `docbook-xml` benutzt werden). Es stellt außerdem Skripte zum Erstellen und Gestalten der Quelle in verschiedenen Ausgabeformaten bereit.

Dokumentation für die DTD kann im Paket `debiandoc-sgml-doc` gefunden werden.

A.8.3 `debian-keyring`

enthält die öffentlichen GPG- und PGP-Schlüssel der Debian-Entwickler. Siehe Abschnitt 3.2.2 und die Paketdokumentation für weitere Informationen.

A.8.4 debian-maintainers

enthält die öffentlichen GPG-Schlüssel der Debian-Betreuer. Siehe http://wiki.debian.org/DebianMaintainer für weitere Informationen.

A.8.5 debview

debview stellt einen Emacs-Modus bereit, um Debian-Binärpakete anzusehen. Dies ermöglicht Ihnen ein Paket zu untersuchen ohne es zu installieren.