Kaynak Temizleme Kılavuzu

Barış Metin, S. Çağlar Onur

19 Mart 2005

İçindekiler

1 Giriş

Kaynak Temizleme süreci yazılımları bir araya getirme ve sorunlarını giderme işine verdiğimiz takma isimdir. Bu süreç sonunda ürün olarak dağıtıma girmeye hazır paketler değil, **Uludağ** tarafından onaylanmış kaynaklar (yazılımlar) çıkartacaktır.

Buna göre iş akışı kabaca söyle tarif edilebilir.

Kaynak Paket ▷ Kaynak Temizleme Süreci ▷ Uludağ Onaylı Kaynak

Kaynak temizleme sürecinin bir *Kuluçka Dağıtım* üzerinde gerçekleşeceği düşünüldüğünde, giriş-çıkış tablosu aşağıdaki gibi olur.

Girdi	Süreç	Çıktı
Orjinal (vanilla) Kaynak + Kuluçka Dağıtım Yamaları	Problemlerin Giderilmesi	Orjinal (vanilla) Kaynak + Kuluçka Dağıtım Yamaları + Uludağ Yamaları

Bu belge kaynak temizleme süreci ne dahil olan her geliştirici için bir kılavuz olmayı amaçlamaktadır.

2 Hazırlık

Kaynak temizleme işi, şu an için, yazılım listemizdeki[1] paketlerin kurulu olduğu bir *Kuluçka Dağıtım* üzerinde yapılmaktadır. Bir geliştiricinin yazılımlardaki sorunları temizlemeye başlamadan önce sistemini bir ***"Temizleme Sistemi"***[2] olarak hazırlaması gerekmektedir. *Kuluçka Dağıtım* uygun seçenekler ile kurularak aşağıdaki şekilde bir ***Temizleme Sistemi*** hazırlanabilir.

2.1 Kuluçka Dağıtım (Gentoo) Kurulumu

Kuluçka Dağıtım kurulumu için HD tarafından hazırlanan `ftp://ftp.uludag.org.tr/pub/pardus/rootfs/` adresinden son Pardus kök dosyası indirilmelidir. Bu dosya Uludağ Geliştiricileri tarafından kaynak temizleme sürecinde kullanılmak üzere oluşturulan *Kuluçka Dağıtım*ı, derlenmiş paketleri ile içerir.

Geliştirici sabit diskinde uygun bölümlemeyi yapmalı ve bu bölüme Pardus kök dosyasını açmalıdır. Sistem yeniden başlatılmadan ve *Uludağ Kuluçka Tabanı* kullanılmaya başlamadan önce ***/etc/fstab*** dosyası disk bölümlerine göre düzenlenmeli ve sistemde bulunan **önyükleyici** tekrar ayarlanmalıdır. Kök

[1] Uludağ 1.0 için yazılım seçimleri, *Uludağ 1.0 Yazılım Seçimleri*dökümanında açıklanmıştır.

[2] Temizleme Sistemi, geliştiricinin Uludağ onaylı kaynak paket hazırlamak için kullanacağı sistemi ifade eder.

dosya arşivi içerisinde **grub** *önyükleyicisi* bulunur. *Grub* yapılandırma dosyası ***/boot/grub/grub.conf*** dosyasıdır.

Uludağ Kuluçka Tabanı için öntanımlı root kullanıcısı şifresi ***"uludag"*** dır.

İdeal ***Temizleme Sitemi*** uludag-kulucka arşivi kullanılarak kurulmuş sistemdir. Fakat herhangi bir **Gentoo** kurulumu da aşağıdaki adımlar uygulanarak bir ***Temizleme Sistemi***'ne çevirilebilir.

2.2 Uludağ Paketler deposu

Paketler deposu, *kuluçka dağıtım* deposunun **Uludağ** 1.0 paketlerinin bir kısmını içeren küçük bir kopyasıdır. Paketler deposu kaynak paketlere dair **Uludağ** tarafından yapılan tüm iyileştirmeleri içerir. Temel bir **Gentoo** kurulumu yaptıktan hemen sonra geliştiricinin paket temizlemeye başlayabilmesi için ***https://svn.uludag.org.tr/paketler/*** adresinden **Uludağ** kaynak paket deposu indirmesi gerekir. Depoyu indirmek için;

$ svn co https://svn.uludag.org.tr/paketler

komutunu vermek yeterlidir. Komut, bulunduğunuz dizinde paketler isimli yeni bir dizin oluşturacaktır. Yeni dizin **Uludağ** paketler deposunun bir kopyasıdır. Bu kopyaya çalışma kopyası ismi verilir. Kurulumun bundan sonraki kısmında **Uludağ** kaynak paketler deposu size yardımcı olacaktır.

2.3 make.conf

Depo içerisindeki **make.conf** dosyası **Uludağ** yazılım seçimlerine ve bu yazılımların özelliklerine göre bir **Gentoo** kurulumu için kullanılmalıdır. **make.conf** dosyasını kullanmak için aşağıdaki komut ile dosyayı uygun yere kopyalamak yeterlidir.

$ cp paketler_dizini/trunk/make.conf /etc/make.conf

Bu dosya içerisinde uygun derleme seçenekleri de dahil olmak üzere, ***Temizleme Sistemi***'nin ve **Uludağ** kaynaklarının kurulumu için gerekli seçimler yapılmıştır. Buradaki değişkenlerin anlamları şöyledir;

- **USE:** Sisteme kurulacak olan paketlerin hangi özelliklerinin istenip, hangi özelliklerinin istenmediğini belirten değişkendir.
- **CHOST:** İşlemci mimarisini belirten değişkendir. [P4 için i686, P3 için i586, 486 için i486, i386 için i386 gibi]
- **CFLAGS:** gcc için derlenecek paketlerin hangi derleyici paremetreleri ile derleneceğini belirten değişkendir.
- **CXXFLAGS:** g++ için derlenecek paketlerin hangi derleyici paremetreleri ile derleneceğini belirten değişkendir.

- **PORTAGE_TMPDIR:** Derlenecek paketin açılacağı, derleneceği geçici disk alanını belirten değişkendir.
- **PORTDIR:** Portage ismi verilen paket deposunun nerede olduğunu belirten değişkendir.
- **DISTDIR:** Çekilen kaynak kodlarının ve büyük yamaların nerede olduğunu belirten değişkendir.
- **PKGDIR:** Eğer mevcut ise binary hale getirilip paketlenmiş dosyaların nerede olduğunu belirten değişkendir.
- **PORT_LOGDIR:** Her paket için ayrı bir log dosyasının nerede saklanacağını belirten değişkendir. [*Kapatılabilir*]
- **PORTDIR_OVERLAY: Gentoo** paket deposunun yanında yerel olacak tutulacak paket deposunun yerini belirten değişkendir. [*bkz: 2.4*]
- **MAKEOPTS:** Aynı anda kaç adet *make* sürecinin çalıştırılacağını belirten değişkendir. [*işlemci sayısı + 1 yeterlidir.*]
- **FEATURES:** Portage ismi verilen paket yöneticisinin hangi özelliklerinin kullanılacağını belirten değişkendir.
- **PORTAGE_BINHOST:** Binary hale getirilmiş paketleri çekmek için kullanılacak sunucunun adresini belirten değişkendir.
- **GENTOO_MIRRORS:** Kaynak kodlarının çekilmesi için kullanılacak sunucunun adresini belirten değişkendir.
- **SYNC: Gentoo** paket deposunun senkronlanmasi işlemi için kullanılacak sunucunun adresini belirten değişkendir.

2.4 Uludağ paket deposunun kuluçka dağıtımda kullanımı

Gentoo üzerinde **Uludağ** kaynaklarını kullanabilmek için make.conf dosyası içerisindeki *PORTAGE_OVERLAY* değişkeninin tanımlanması gerekmektedir. *PORTAGE_OVERLAY* değişkeni **Uludağ** kaynak paket deposunun çalışma kopyasını işaret ederek depo içerisindeki kaynakların kullanılmasını sağlar. Bu sayede kuluçka dağıtım paket sistemi öntanımlı olarak **Uludağ** deposunu kullanacaktır.

Eğer **Uludağ** paketler deposunu /home/user/paketler dizinene çektiyseniz; **/etc/make.conf** dosyası içerisine

PORTAGE_OVERLAY=/home/user/paketler/trunk

satırını eklemek yeterlidir.

2.5 Yazılımlar

Uludağ yazılımları *temel yazılımlar* ve *uygulama yazılımları* olmak üzere iki sınıfa toplanmıştır.

Bu sınıflar için uludag-base (temel yazılımlar için) ve uludag-apps (uygulama yazılımları için) ebuild dosyaları Uludağ paketler deposuna yerleştirilmiştir.

Uludağ Temel yazılımlarının kurulumu için aşağıdaki komut yeterlidir.

 # emerge uludag/uludag-base

Uygulama yazılımları için ise

 # emerge uludag/uludag-apps

komutu kullanılır. Temel ve uygulama yazılımlarını tümü ile kurmak için ise

 # emerge uludag/uludag-all

komutu kullanılır.

Uludağ tarafından seçilen yazılımların listesine depo içerisindeki *repos/docs* dizininden ulaşılabilir.

Emerge ile uludag paketlerini kurmaya başlamadan önce her paket için *digest* ve *Manifest* dosyalarının oluşturulması gerekir. **ebuild**, *digest* parametresi ile, bu iş için kullanılır. Deponun trunk/ dizininde, depodaki her paket için bu komutu çalıştırmayı kolaylaştıran bir Makefile dosyası bulunmaktadır. İşlemi gerçekleştirmek için trunk dizininde;

 # make ebuild_all_digest

komutunu vermeniz yeterlidir. Bu işlem tüm paketler için kaynak kodları da internetten indireceğinden uzun sürecektir.

2.6 Türkçe yapılandırması

Türkçe yapılandırma işlemleri, uludag-*.tar.bz2 arşivi kullanılarak yapılan kurulumlarda hazır olarak gelir ve tekrar yapılandırma gereksizdir. Uludag arşivini kullanmayan kurulumlarda yapılandırma seçeneklerinin yukarıda anlatıldığı gibi düzenlenmesi gerekir.

Temizleme Sistemi'nin Türkçe UTF-8 ayarları ile yapılandırılması ve kullanılması gerekmektedir. Konsolda ve grafik ekranda Türkçe çalışabilmek için öncelikle yerel değişkenlerinin Türkçe yapılması amacıyla **LC_ALL** ve **LANG** yerel değişkenleri **tr_TR.UTF-8** olarak tanımlanmalıdır.

Bu yerel değişkenlerini tanımlamak için **/etc/env.d** dizini altına *03locale* adında bir dosya oluşturarak içerisine aşağıdaki satırları yazabilirsiniz.

LC_ALL=tr_TR.UTF-8
LANG=tr_TR.UTF-8

2.6.1 Metin ekranda Türkçe desteği

Metin ekranda Türkçe desteği için Türkçe klavye haritası *trqu.map.gz* ve Türkçe font dosyası *iso09.16.gz* kullanılmalıdır.

Klavyenin unicode modunda çalışması için"*kbd_mode -u*" komutu kullanılır. Bu özelliği etkin hale getirmek için **/etc/rc.conf** dosyasında KEYMAP değeri aşağıdaki şekilde düzenlenir.

KEYMAP="trq" # Türkçe klavye haritası
UNICODE="yes"
CONSOLEFONT="iso09.16.gz" # Türkçe font
CONSOLETRANSLATION="8859-9"

Terminalde unicode desteği için ise terminale"%G" escape karakteri *echo -e"\033%G"* komutu ile gönderilir. Bunun için **/etc/issue** dosyası kullanılabilir. Dosyayı uygun içerik ile oluşturmak için basitçe aşağıdaki komut kullanılabilir.

 # echo -e "\0332J\033[f\033%GUludağ Linux'a Hoşgeldiniz" > /etc/issue

2.6.2 Grafik Ekranda Türkçe desteği

Grafik ekranda Türkçe desteği için xorg.conf yapılandırma dosyasında *Option "XkbLayout" "tr"* satırı ile Türkçe klavye desteği verilir.

Masaüstü olarak kullanılan KDE'de Türkçe arayüz için kde-i18n paketinin *LINGUAS=tr* parametresi ile kurulması gerekmektedir. Aşağıdaki komut bunun için yeterlidir.

 # LINGUAS=tr emerge kde-i18n

3 Araçlar ve Kullanımları

Kurulum ve temel hazırlıktan sonra paket temizlemeye geçilebilir. Bunun için kullanılan araçların tanımları ve kullanımı aşağıda anlatılmıştır.

3.1 Bugzilla

Uludağ Bugzilla'sında paketler için ayrı bir kategori bulunur. Paketler kategorisi altına her paket ayrı bir bileşen(component) olarak yalnızca paket isimleri ile eklenir.

Örneğin: bash-2.05b-r9 kuluçka dağıtım paketi, sürüm numarası olmadan, yalnızca *bash* ismi ile depoya eklenir. Depoya, paket sürüm numarası yerine paketin dağıtımın hangi sürümü için hazırlanmış olduğu bilgisi girilir. Şu an işlettiğimiz paket temizleme sürümü için bu 1.0_pre sürümünü ifade eder.

3.2 Subversion

Kurulum sırasında kullanılan **Uludağ** Kaynak Paketler Deposu her ***Temizleme Sistemi'***nde zaten hazır olarak bulunur. Depodaki paketler kuluçka dağıtımın sunduğu gruplamalara uygun bir şekilde kendi dizinlerine yerleştirilmiştir. Bu sayede, kurulum bölümünde anlatıldığı gibi, depodaki paketleri alarak tümü ile *Kuluçka dağıtım* üzerinde kullanabilme olanağı olur. Ayrı bir depo üzerinde çalışmak bize *kuluçka dağıtım* ile karışmadan fakat aynı sistem üzerinde gelişmeye ve evrimleşmeye uygun ortamı sağlar.

3.2.1 Güncel çalışma kopyası

Depo'dan bir çalışma kopyasının nasıl alınabileceğini zaten biliyoruz. Fakat, güncel kaynaklar ile çalışabilmek için geliştirici her çalışma öncesi paket deposunu güncellemelidir. Bunun için aşağıdaki komutları vermek yeterlidir. Aşağıda adı geçen $ULUDEPO yerel değişkenlerini tanımlamak için **/etc/env.d** dizini altına *33depo* adında bir dosya oluşturarak içerisine *ULUDEPO="Uludağ paketler deposunu yolu"* satırını aşağıdaki gibi yazabilirsiniz.

ULUDEPO=/home/user/paketler/trunk

$ cd $ULUDEPO

$ svn update

3.2.2 Depo yapısı

Depo içerisinde üç ana dizin bulunur; trunk, tags ve repos.

- **trunk** dizini sürekli çalışmanın yapıldığı dizindir. Geliştirici yaptığı ve test ettiği tüm yamaları öncelikli olarak trunk dizini altındaki kaynak paket deposuna yerleştirir.
- **tags** dizini kaynak paketlerin belirli bir olgunluğa geldiğinde kopyalandıkları dizindir. Bir kaynak paketin sorumlusu (maintainer) kaynak paketin kararlılığından emin olduğunda, pakete bir sürüm numarası vererek taglar(işaretler).
- **repos** dizini geliştiricinin kaynak paketler üzerindeki deneysel çalışmalarını yaptığı dizindir. Geliştirici repos altına kendi dizinini oluşturarak bu dizin içerisinde mevcut kaynak paketlere veya yeni bir yazılıma dair deneysel çalışmasını yürütebilir. Repos dizini aynı zamanda kaynak temizleme dökümanlarının da içerisinde bulunduğu *docs* dizinini de barındırır.

Uludağ kaynak paket deposunda seçilen kuluçka dağıtıma uygun şekilde yerleştirilmiş paket kurma betikleri ve kuluçka dağıtım tarafından yapılan yamalar ve öntanımlı ayar dosyaları bulunmaktadır. Paketlerin kurulması için gerekli olan bazı dosyalar dinamik olmaları sebebi ile depoya yerleştirilmemişlerdir. Bu dosyalar paket kurma işlemi sırasında ebuild yazılımı tarafından oluşturularak kullanılacaklardır.

Depo içerisindeki her kaynak paket dizininde aşağıdaki dosyalar bulunur.

1. *Gentoo ebuild dosyası*: Bu dosya portage tarafından kullanılan derleme/kurulum dosyasıdır. Yazılıma yapılan değişiklikleri uygun bir şekilde test etmek için kaynak paket sorumlusu ebuild dosyasında da bazı güncellemeler yapmaktadır.

2. *Yamalar*: **Gentoo** yamalarının 20 Kb'tan küçük olan ve **Uludağ** tarafından yapılan yamaların tümü **files** dizini altında bulunur. 20 Kb'tan büyük olan yamalar **distfiles** klasöründe sıkıştırılmış halde tutulur. Kaynak sorumlusu bu dosyayı almalı ve paketin **files** klasörüne açmalıdır.

3. *Changelog* dosyası: Uludağ geliştiricileri tarafından yapılanların kaydı bu dosyada tutulur. Bu dosya kaynak sorumlusu tarafından ilk ihtiyaç duyulduğunda oluşturulur.

4. *Takip* dosyası: Takip dosyası kaynak geliştiricisi tarafından kaynak ilk ele alındığında hazırlanır. Takip dosyası içerisinde kaynak sorumlusu(maintainer) ve kaynağın o sıradaki durumu listelenir. Örnek takip dosyasını yapısı şu şekildedir.

PAKET=<paket adı>
VERSIYON=<versiyon numaras>
SORUMLU=<eposta adresi>
BUGZILLA=< Uluzilla URL'si >
CEVIRI=< % cinsinden alınan yol >
UTF8=< % cinsinden alınan yol >
YAPILANDIRMA=< % cinsinden alınan yol >
GRAFIK_MEDYA=< % cinsinden alınan yol >
HATALAR=< % cinsinden alınan yol >
KULLANIŞLILIK=< % cinsinden alınan yol>

3.2.3 Yapılanların gönderilmesi

Uludağ kaynak paket deposunu güncelledikten sonra, geliştirici kaynaklar üzerinde çalışmaya başlayabilir. Kaynaklara dair yapılan tüm güncellemenin Uludağ deposu üzerinden yapılması zorunludur.

İşi biten geliştirici aşağıdaki komut ile yaptıklarını ana depoya gönderebilir.

$ svn commit

Komuttan hemen sonra gelen yorum ekleme ekranı yapılan işin tarif edilmesi içindir. Yapılan işin burada açıklayıcı bir şekilde tarif edilmesi diğer geliştiricilerin yaptıklarımızdan haberdar olmasını sağlar. Eğer aynı zamanda üzerinde çalıştığınız kaynak paketin sorumlusu iseniz kaynak paket dizinindeki *ChangeLog* dosyasına da yapılanları yazmanız gerekir.

Paket deposuna yazma hakkı yalnızca geliştiricilere aittir. Fakat herkes depoyu kullanım için çekebilir. Depoya yazma hakkı olmayan geliştiriciler kaynak paketler üzerinde yaptıkları çalışmaları kaynak paketin sorumlusuna göndererek çalışmalarının depoya girmesini sağlayabilirler.

3.3 Portage: emerge, ebuild

ebuild, hem seçilen *kuluçka dağıtım*ın **portage** ismi verilen paket yöneticisinin metodlarına direk erişmek için kullanılan arayüze verilen isim hemde paket veritabanını oluşturan kurulum betiği dosyalarının adıdır. *ebuild* yazılımı aşağıdaki metodlara sahiptir < *ebuild* **paketadı *metod*** >;

- **setup:** Paket için gerekli tüm sistem gerekliliklerini kontrol eder.
- **fetch:** Paket'in kaynak kodunu ve yamalarını çeker.
- **digest:** Paketin kaynak kodu ve yamalarının md5sumlarını Manifest ve digest-paketismi dosyalarına yazar.
- **unpack:** Paketin kaynak kodunu */var/tmp/portage/paketadı* klasörüne açar ve yamaları yapar.
- **compile:** Paketin yamaları uygulanmış kaynak kodunu derler.
- **preinst:** Paketi gerçek sisteme kurulmadan önce gerekli işlemleri yapar.
- **install:** Paketi kendi sandbox'ının içine kurar.
- **postinst:** Paket gerçek sisteme kurulduktan sonra gerekli işlemleri yapar.
- **qmerge:** Sandbox'a kurulan paketi gerçek dosya sistemine kurar.
- **merge:** Sırası ile **fetch, unpack, compile, install** ve **qmerge** süreçlerini işletir.
- **unmerge:** Sisteme kurulmuş paketi kaldırır.
- **prerm:** Sisteme kurulmuş paket kaldırılmadan gerekli işlemleri yapar.
- **postrm:** Sisteme kurulmuş paket kaldırıldıktan sonra gerekli işlemleri yapar.

- **package:** Seçilen paket için **fetch, unpack, compile, install** şüreçlerini işlettikten sonra dosya sistemine kurmak yerine derlenmiş halini .tbz2 dosyası haline getirir.
- **rpm:** Seçilen paket için **fetch, unpack, compile, install** şüreçlerini işlettikten sonra dosya sistemine kurmak yerine derlenmiş halini .rpm dosyası haline getirir.

Bir *ebuild* dosyası ise içerisinde yukarıdaki metodları sağlayacak fonksiyonları ve çeşitli değişkenleri barındırır. *ebuild* arayüzü bu dosyayı işleyerek sisteme kurulacak paket için gerekli işlemleri gerçekleştirir.

Emerge ebuild arayüzünü kullanan **Gentoo**'nun paket yöneticisine verilen isimdir. Paket kurmak, kaldırmak, özelliklerini yada bağımlılıklarını listelemek gibi işlevleri *ebuild* arayüzünü kullanarak yerine getirir. *emerge* ile bir paketi kurmadan önce *emerge -lvp paketadı* komutu ile kurulacak olan paketin ve getirdiği bağımlılıkların listesi alınmalı ve sisteme neler getireceği incelenmelidir. Bir paketi sistem kurmak ve eskisini kaldırmak için *emerge paketadı*, paketi sadece kaldırmak için *emerge unmerge paketadı* komutlarını kullanmak yeterlidir.

4 İşleyiş

Yazılımlar ve yazılımlara yapılan değişiklikler http://svn.uludag.org.tr/paketler deposunda saklanmaktadır. Hata bildirimi ve takibi ise ***http://bugs.uludag.org.tr*** adresinden yayın yapan **Uludağ** Bugzilla'sı içerisindeki Paketler kategorisi kullanılarak yapılır. Her kaynak paket sorumlusu kendi paketi için depo ve bugzilla içeriklerini hazırlamak zorundadır.

Kaynak paket deposu dağıtıma girecek tüm yazılımları içerir. Kaynak paket sorumlusu yalnızca eksik olan içeriği oluşturmalıdır.

Sıradan bir kaynak için, kaynak ilk ele alındığında *takip* dosyası ve kaynağa dair yapılan ilk işte *Changelog* dosyasının oluşturulması yeterlidir. Kaynak pakete dair diğer dosyalar zaten depo içerisinde bulunmaktadır.

Yeni bir geliştiricinin hazırlık 2 bölümünde anlatıldığı gibi **Uludağ** deposunu kullanarak bir *Kuluçka Dağıtım* kurulumu yapması ve bu canlı sistem üzerinde sorunların çözümü için çalışması gerekir.

4.1 Takip/Temizleme Süreci

Paketlerin sorunlarının temizleme ve takip süreci aşağıdaki adımların izlenmesinden oluşacaktır.

4.1.1 Sorun tespiti

Kuluçka dağıtım seçilen tüm yazılımları sağlar bir şekilde derleme/test makina(ları)na kurulur ve bu makinalar üzerinde sorunlar (yukarıda listelenen problemler göz önüne alınarak) tespit edilir. Yazılımlardaki sorunları tek tek ele almak yerine **canlı**[3] bir sistem üzerindeki sorunları gözlemleyip çıkarmak öncelikli olarak izlenen yoldur.

4.1.2 Sorunun kaynağının bulunması

Sorun hangi paketten (veya paketlerden) kaynaklanıyor bulunacak. Bu aşamada Bugzilla'ya soruna ilişkin bir giriş yapılacak.

4.1.3 Çözüm araştırması

Sorunun çözümü için araştırma yapılacak. Tüm bulgular Bugzilla'daki hata girişine eklenecek.

4.1.4 Çözüm

Sorunun çözümü bulunduğunda yine Bugzilla'daki hata girişine çözüm önerisi eklenecek.

4.1.5 Yama

Çözümü sağlayan yama yapılacak; yama test edilmek üzere Bugzilla'daki hata girişine eklenecek.

4.1.6 Test

Sorunun çözümü test edilecek ve çalıştığından emin olunacak.

4.1.7 Depoya yerleştir

Çözümü sağlayan yama depoya koyulacak; Bugzilla'daki hata kapatılacak.

[3] **Canlı Sistem**: Dağıtıma girmek üzere seçilen tüm uygulamaları üzerinde çalışır vaziyette bulunduran sistem.

4.2 Temizlenecek problemler

Paket temizleme aşamasında yazılımların şu problemlerini ele alacağız.

1. **Utf-8** sorunları
2. **Türkçe çevirileri**
3. Uygulama **grafikleri**
4. **Ön tanımlı yapılandırma seçenekleri** (yerel ayarları dahil)
5. Yazılımların bilinen ve bizim tarafımızdan bulunan **hataları** (buglar).
6. **Kullanışlılık yamaları**.

4.3 Temizleme Önerileri

Yazılımı değiştiren tüm geliştirme yazılımın ana kaynağı (geliştiricisi) ile etkileşimli olarak yapılmalıdır. Yapılan tüm iyileştirmenin ana kaynağa iletilerek bir sonraki sürümün daha sorunsuz çıkması sonraki çalışmalarda geliştiricinin yazılım üzerindeki iş yükünü azaltacaktır.

Problem tiplerine bakıldığında, her problem tipi için bazı çalışmaların hali hazırda mevcut olduğu görülebilir. Geliştirici problemlerin çözümü için çalışırken mevcut emeği en etkin şekilde kullanmayı amaçlamalıdır.

Paket sorumlusu, sorumluluğunda olan paketler için eğer bir yerelleştirme çalışması yapılmış veya yapılıyor ise bu çalışmaya destek olmalıdır. Bununla birlikte yazılımın utf-8 veya Türkçe yerelleri ile ilgili sorunları da direkt olarak yazılımın ana geliştiricisi ile birlikte çözülmeye çalışılmalıdır. Bunun için tüm problemlerin çözümlerinin aranmasında aşağıdaki adımlar işletilebilir.

1. Ana geliştiricinin hata raporları incelenir. Eğer orada böyle bir hata bildirilmemiş ise ana geliştirici bilgilendirilir.
2. Bulunan hata diğer **Uludağ** geliştiricilerinin haberdar olabilmeleri ve hatanın takibi için ***http://bugs.uludag.org.tr*** 'ye raporlanır.
3. Problemin çözümü ana geliştiricinin tartışma listelerinde aranabilir.

4.4 Yama Nasıl Uygulanır?

Paket temizleme sürecinde bir geliştiricinin yeni bir yamayı depoya eklerken izleyeceği ideal yol aşağıdaki adımlarla tarif edilebilir.

4.4.1 Kaynak kodu al/aç.

Gentoo'da ana geliştiricinin kaynak paketleri ön tanımlı olarak */usr/portage/distfiles* altına kopyalanır. Geliştirici, yazılımın kaynak kodu ile çalışmak istediğinde bu dizindeki kaynak kod dosyasını uygun bir yere açar.

Eğer üzerinde çalışılmak istenen kaynak paket *distfiles* dizini içerisinde bulunmuyorsa,

 # emerge -f paket_adi

komutu ile yazılımın kaynak paketinin ve varsa binary halde gelen yamalarının indirilmesi sağlanabilir. Kaynak kodun varsa yamaları ile birlikte açılmış halini ebuild yazılımı aracılığı ile */var/tmp/portage/paketadı* klasörüne açmak için

 # ebuild paketadı unpack

komutu kullanılabilir.

4.4.2 Yamayı hazırla

Kaynak kod üzerinde gerekli işlemler yapıldıktan sonra, geliştirici yaptığı düzenlemeleri bir yama olarak hazırlayabilmek için başka bir dizine ana geliştiricinin kaynak kodunu tekrar açar ve yaptığı düzenlemelerin farkını alır. Fark alma işlemi için standart Unix aracı diff, -u parametresi ile kullanılır.

$ diff -u kaynak_dizini_temiz kaynak_dizini

4.4.3 Yamayı çalışma kopyasına yerleştir.

Geliştirici tarafından üzerinde çalışılan paketin sorunlarını çözmek amacı ile uygulanan yamalar çalışma kopyasındaki paketin kendi klasöründe bulunan **"files"** klasörünün içine uygun bir isim ile yerleştirilir [örneğin; paketadı-versiyonu-sorun.patch gibi].

4.4.4 Test et.

Üzerinde çalışılan paketin ebuild dosyası bir metin düzenleyici ile açılır. ebuild'in ***src_unpack()*** fonksiyonunun [4] *uygun yerine;*

- ***http://www.gentoo.org/proj/en/devrel/handbook/handbook.xml?part=2&chap=1*** adresinde bulunan bilgiler doğrultusunda ***"epatch ${FILESDIR}/${P}-sorun.patch"*** satırı

[4] Ebuild'lerin bazıları bu kurala göreceli olarak uyabilir. Örneğin glibc için yamalar sayıca fazla olduğundan direkt yama satırını eklemek yerine geliştiriciler bu yama setlerini fonksiyon haline getirmiş ve src_unpack içinden çağırmayı tercih etmişlerdir.

yada,

- **“epatch files/paketadı-versiyonu-sorun.patch”** satırı

eklenir.

Gerekli yama ebuild dosyasına eklendikten sonra paketin kendi klasörü içinde ***“ebuild paketadı.ebuild digest”*** komutu çalıştırılarak paketin kurulması için gerekli olan ***Manifest*** ve ***digest-**** dosyaları oluşturulur. Bu noktadan sonra geliştirici paketi ***“emerge paketadı”*** komutu ile sisteme kurabilir.

4.4.5 Ana depoya ekle.

Yama geliştiricinin testlerini geçtikten sonra depo içerisine (svn commit ile) eklenir.

Yama depoya gönderilmeden önce mutlaka çalıştığından emin olunmalıdır. Depo her ne kadar sürekli olarak güncelleniyor olsa da, tüm geliştiricilerin depo ile sorunsuz bir şekilde çalışabilmeleri için depodaki kaynakların her zaman çalışıyor olması gerekir.

Başka bir kişinin sorumluluğunda olan bir yazılım için yama hazırlandığında, ekleme işleminden önce kaynak paket sorumlusunun onayı alınmalıdır.

4.4.6 Bugzilla’da ilgili hata girişi kapat.

Probleme dair uygun yama test edildiğinde geliştirici daha önce problem için açılmış olan bugzilla’daki hata girişini kapatır.

4.4.7 Changelog’a yapılan değişiklik yazılır.

Changelog dosyasına yapılan işlemler kısa birer açıklama ve bugzilla’daki hata numarasına referans verilerek eklenir.

4.4.8 Ana geliştirici yapılan iyileştirmeden haberdar edilir.

Bundan sonra geliştirici, bir sonraki sürümün bu iyileştirmeyi içerebilmesi için, ana geliştiriciyi yamadan haberdar etmelidir.

4.5 Yazılım Seçimi ve Sürüm Güncellemeleri Politikası

Paketler deposu, *kuluçka dağıtım* deposunun **Uludağ** 1.0 paketlerini içeren küçük bir kopyasıdır. Buradaki yazılım seçimleri yapılmış ve depoya yerleştirilmiştir. Depoda olan paketlerin daha güncel kopyaları ile değiştirilmesi kararları paket sorumlusunun insiyatifindedir, fakat bu güncel kopyalar seçilen *Kuluçka Dağıtım*'ın kararlı paketleri arasından olmalıdır. Eğer paket versiyonunun değişmesi birden fazla paketi etkiliyor ise (örneğin glibc) bu karar geliştiriciler arasında varılacak karara bağlıdır.

Kaynak paket sorumlusu bir yazılımın sürüm numarasını güncellediğinde uludag/uludag-base veya uludag/uludag-apps ebuild dosyasındaki sürüm numarasını da güncellemelidir.