UNIVERSIDADE FEDERAL DE SANTA CATARINA

ESTUDO DE IMPLEMENTAÇÃO DE UMA DISTRIBUIÇÃO GNU / LINUX

CRINEU TRES
ULISSES MUNIZ DE QUEIROZ

UNIVERSIDADE FEDERAL DE SANTA CATARINA
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA
CURSO DE CIÊNCIAS DA COMPUTAÇÃO

ESTUDO DE IMPLEMENTAÇÃO DE UMA DISTRIBUIÇÃO GNU / LINUX

Crineu Tres

Ulisses Muniz de Queiroz

Professor Orientador: José Eduardo de Lucca

Florianópolis, fevereiro de 2004

CRINEU TRES
ULISSES MUNIZ DE QUEIROZ

ESTUDO DE IMPLEMENTAÇÃO DE UMA DISTRIBUIÇÃO GNU / LINUX

Trabalho de conclusão de curso em
Bacharelado em Ciências da Computação
apresentado como requisito para a obtenção
do Grau de Bacharel em Ciências da
Computação.

Orientador: Prof. José Eduardo De Lucca

Florianópolis, fevereiro de 2004

CRINEU TRES
ULISSES MUNIZ DE QUEIROZ

ESTUDO DE IMPLEMENTAÇÃO DE UMA DISTRIBUIÇÃO GNU / LINUX

Este trabalho de Conclusão de Curso foi julgado adequado para a obtenção da Graduação em Bacharelado em Ciências da Computação.

Apresentado à Banca Examinadora, integrada por:

José Eduardo de Lucca – Professor Orientador

André Luis Gobbi Sanches

Augusto César Campos

Florianópolis, fevereiro de 2004

A todos aqueles que nos apoiaram.

Agradecimentos

Ao noso orientador De Lucca e ao membro da banca Gobbi, pelas idéias e pelo incentivo dado ao trabalho, além da disponibilização dos equipamentos do laboratório.

Ao ex-colega de trabalho e membro da banca Augusto Campos, por todas sugestões dadas, até aquelas que não foram seguidas e serviram para futuro arrependimento.

Aos colegas de turma Bruno Leonardo Martins de Melo e Túlio Vinicius Duarte Christofoletti, pela seriedade e maturidade demonstradas ao longo do curso.

Também aos colegas Vítor Claudino dos Santos, Roberto Hartke Neto, Marcel de Castilho, Igor Tibes Ghisi, Henrique Viecili, Diogo Fernando Veiga e Cláudio Ulisses Nunes Biava, que sempre estiveram acordados e presentes nas horas em que precisávamos.

Aos mestres Aldo von Wangenheim, Antônio Augusto Medeiros Fröhlich, Mário Antônio Ribeiro Dantas, José Francisco Danilo de Guadalupe Correa Fletes, Julio Felipe Szeremeta, Olinto José Varela Furtado e Sérgio Peters, pela insistência em tentar nos transferir algum conhecimento.

E aos amigos Diego Votcoski, Marcelo Zannin da Rosa, Paula Cristina de Aguiar, Paulo Bruno Balvedi, Priscilla Taís Bueno e Ronaldo Batalha Silva, que sempre acreditaram no nosso potencial.

“I know it's important, I honestly do but we're talking about practice. We're talking about practice man. We're talking about practice. We're talking about practice. We're not talking about the game. We're talking about practice.”

Allen Iverson

Resumo

Este trabalho se propõe a criar uma distribuição GNU / Linux para ser usada em praticamente qualquer computador e que possa ser adquirida de forma gratuita, além de possui todas funcionalidades de um Sistema Operacional proprietário.

Uma distribuição base foi escolhida e então modificada para ficar de acordo com o desejado. Algumas das modificações mais importantes dizem respeito à diminuição do tamanho, disponibilização dos softwares na língua nativa e inclusão de aplicativos considerados convenientes.

Palavras-chave: Linux, Distribuição GNU / Linux.

Abstract

The purpose of this work is to create a GNU / Linux distribution to be used in practically any computer and acquired at no financial cost, besides having all features of a proprietary Operational System.

A base distribution was chosen and then modified to be according the project. Some of the major modifications are about size restrictions, native language software availability and inclusion of helpful software.

Keywords: Linux, GNU / Linux distribution.

Sumário

1 Introdução

Embora a criação do primeiro computador tenha ocorrido há mais de 50 anos, apenas nas últimas décadas do século XX que este pequeno aparelho conquistou a onipresença que possui atualmente e sua difusão como ferramenta de trabalho, estudo e entretenimento. O chamado Computador Pessoal – ou PC (Personal Computer) – foi responsável por uma das maiores transformações culturais da história da humanidade.

Juntamente com a popularização dos PCs veio a popularização dos programas neles usados. Um em especial esteve intrinsecamente ligado com o crescimento do próprio PC: trata-se do Sistema Operacional Windows, da Microsoft, que por muito tempo dominou praticamente sem concorrência o mercado dos desktops (computadores domésticos).

Esse cenário começou a mudar em meados da década de 90, quando o sistema GNU / Linux deixou de ser exclusividades de servidores e centros de pesquisa e invadiu os desktops. Embora inicialmente não fosse um sistema pronto para o usuário doméstico, ganhou espaço aos poucos e começou a conquistar os usuários mais experientes.

A partir deste primeiro contato com o mundo dos desktops as distribuições GNU / Linux passaram a concorrer diretamente com outros Sistemas Operacionais específicos para este nicho do mercado, e muitas foram criadas especialmente para ele.

Desde essa época, a grande batalha está em convencer o usuário de que o GNU / Linux não é um sistema operacional que necessite de conhecimentos profundos para ser utilizado. É preciso vencer também a relutância das pessoas em mudar de Sistema Operacional, após o costume criado por anos utilizando o mesmo produto. Essa busca por amigabilidade e aceitação é o principal passo para colocar o GNU / Linux à altura de seus concorrentes, eliminando as diferenças técnicas – que representavam uma grande barreira para sua aceitação – e deixando o usuário com o poder de decidir entre um ou outro Sistema Operacional por opção própria.

1.1 APRESENTAÇÃO

Distribuições GNU / Linux, sistema GNU e o kernel Linux geralmente são confundidos entre si, e em muitas ocasiões méritos e deméritos de um e de outro são concedidos ao elemento errado.

A história do Linux começa no início da década de 90, quando o então estudante finlandês Linus Torvalds resolveu criar um Sistema Operacional melhor do que o Minix – sistema baseado no UNIX, porém muito mais simples, concebido por Andrew Tanenbaum com fins didáticos. O trabalho de Torvalds evoluiu para tornar-se o que hoje conhecemos por Linux, o kernel da nossa distribuição.

O kernel é o núcleo do sistema operacional, isto é, o responsável pelas tarefas mais importantes realizadas em um computador, como gerência de memória, gerência de processos e manipulação do sistema de arquivos. Segundo [11]:

> A arquitetura de um sistema operacional corresponde à imagem que o usuário tem do sistema, a forma como ele percebe o sistema. Essa imagem é definida pela interface através da qual o usuário acessa os serviços do sistema operacional. Essa interface, assim como a imagem, é formada pelas chamadas de sistema e pelos programas de sistema.

As chamadas de sistema são implementadas pelo kernel, e é através delas que os programas de sistema se comunicam com o sistema operacional, requisitando serviços necessários à sua execução. O Linux é, portanto, o responsável pela implementação e realização de todos os serviços necessários para habilitar a execução dos programas de sistema.

O sistema GNU – GNU's Not Unix (GNU não é UNIX) – tem seu início um pouco antes. No final de 1983 Richard Stallman decidiu criar um sistema operacional inteiramente novo e free, baseado no UNIX, porém não idêntico – daí o significado da sigla GNU. Sua pretensão inicial era conceber um kernel e utilitários para criar e rodar programas em C: um editor, um shell, um compilador C, um ligador e um assembler.

No início dos anos 90 o projeto GNU estava quase pronto, com exceção de seu kernel, o GNU Hurd, cujo desenvolvimento estava sendo mais difícil do que planejado. Mas não foi preciso esperar a finalização do kernel GNU, pois nessa época o kernel Linux já era uma realidade. O que houve então foi o trabalho de união entre o kernel de Torvalds e o sistema de Stallman, resultando em um sistema

operacional completo e free, primeiramente denominado “a Linux-based version of the GNU system” (uma versão do sistema GNU baseada em Linux) e mais tarde simplificado para somente GNU / Linux.

Uma distribuição GNU / Linux, por sua vez, é o aglomerado de diversos aplicativos rodando sobre o Sistema Operacional GNU / Linux. Tais aplicativos geralmente são programas de sistema, aplicativos de Internet, de escritório, de manipulação gráfica etc.

Embora todas as distribuições GNU / Linux fundamentem-se no mesmo âmago, as diferenças essenciais encontram-se na documentação e nos aplicativos oferecidos por cada uma. As distribuições majoritárias geralmente contam com programas próprios para realizar a instalação do sistema e também para verificar a disponibilidade e, se for o caso, realizar a instalação de atualizações dos componentes do sistema. Em muitos casos as atualização tornam-se disponíveis quando uma falha de segurança é detectada em algum componente do sistema. Atualizações que trazem novas funcionalidades geralmente são incluídas nas versões subseqüentes da distribuição.

Certas características são fundamentais para o sucesso de uma distribuição, e tais características são definidas de acordo com o público alvo. Distribuições que visam o mercado de servidores empresariais devem contar com um grande número de aplicativos deste tipo, tais como servidores de mail, de arquivos, de banco de dados etc. Nesse caso, determinados aspectos, como a estabilidade, robustez, segurança e desempenho, são mais importantes que outros, como a usabilidade e a beleza. Já em uma distribuição voltada para o uso doméstico por pessoas consideradas leigas – como a proposta por esse trabalho – deve zelar pela facilidade do uso, simplicidade, comodidade e uma boa aparência visual, de modo a agradar o usuário.

1.2 JUSTIFICATIVA

Atualmente o governo brasileiro atravessa, assim como governos de diversos países, um processo de revisão dos gastos com software proprietário. Nesse contexto, a adoção do software livre aparece como principal alternativa já que as tarefas realizadas por softwares proprietários podem ser igualmente realizadas

com um custo menor.

A utilização de software livre não tem apenas motivação econômica, mas também social. O baixo custo é uma peça chave na promoção da inclusão digital. A inclusão digital pode ser vista como uma alfabetização eletrônica, daí conclui-se que os principais beneficiados por este processo são pessoas com pouco ou nenhum conhecimento em informática.

As necessidades de um usuário iniciante são bem diferentes das necessidades de um usuário experiente. Para um usuário iniciante, é importante que todas as tarefas sejam fáceis de executar. Alguns cuidados devem ser tomados para melhorar o aprendizado, como por exemplo, evitar que muita informação seja apresentada de uma só vez e que termos técnicos sejam utilizados sem alguma explicação prévia. Enquanto um usuário avançado prefere ter várias opções para realizar uma determinada tarefa, um usuário iniciante prefere ter apenas a mais compreensível.

A idéia de um pacote contendo todas as aplicações desejadas é bastante interessante para facilitar e agilizar os procedimentos de instalação e de manutenção de um sistema, abstraindo boa parte da complexidade com a qual este processo contaria.

O processo de aprendizagem também torna-se muito mais fácil e acessível quando a informação é apresentada na língua nativa do aprendiz, no caso o português do Brasil. O uso de uma língua estrangeira adiciona uma camada de complexidade ao aprendizado, e isola uma grande parte do público-alvo.

Um sistema de baixo custo não significa necessariamente um equipamento velho e obsoleto, mas em muitas vezes este é o caso; portanto a distribuição deve ser capaz de rodar em equipamentos modestos, para torná-la mais abrangente. Isso implica que poucos recursos de memória principal e secundária devem ser pretendidos.

Levando em conta a parcela da população a qual a distribuição é destinada é imprescindível que somente software livres sejam utilizados, de forma a não gerar custos extras.

Uma distribuição GNU Linux em português do Brasil, de fácil instalação, com aplicações de Internet / escritório e com um menu de navegação simples é uma solução apropriada para adotar-se em iniciativas de inclusão digital.

Existem várias distribuições com algumas das características citadas

acima, mas nenhuma que possua todas. Em muitos casos as aplicações aqui consideradas fundamentais – como o pacote de aplicativos para escritório OpenOffice – estão presentes em português de Portugal. O usuário geralmente precisa tomar muitas decisões durante o processo de instalação, e o número de aplicativos disponíveis é bastante elevado, causando ainda mais confusão. Além disso, alguns aplicativos desnecessários para os propósitos aqui definidos são instalados, como servidores de e-mail e agendadores de tarefas, que consomem recursos de processamento e memória e confundem os usuários menos experientes.

2 Objetivos

O objetivo principal deste trabalho é a criação de uma distribuição GNU / Linux ideal para primeiro contato entre principiantes e computadores.

2.1 GERAIS

De forma ampla, podemos definir que o trabalho visa analisar as distribuições GNU / Linux existentes na procura de alguma que contenha a maior parte das características cobiçadas e modificá-la a fim de atingir os objetivos propostos.

A distribuição precisa ser capaz de atender as necessidades de usuários leigos que possuem um computador, pouco ou nenhum conhecimento de informática e não desejam adquirir softwares proprietários para realização de tarefas básicas como acesso à Internet e edição de textos.

É importante fazer o máximo possível para simplificar a vida do usuário. As tarefas que permitirem devem ser automatizada de forma que o usuário não necessite realizá-las ou se abstenha de tomar decisões para concluí-las. Um Sistema Gerenciador de Janelas simples e amigável deve ser adotado, na intenção de agilizar a adaptabilidade do usuário ao novo sistema.

O resultado final deve conter uma distribuição simples e eficiente, apenas com o necessário para a realização das tarefas básicas propostas e um mecanismo de instalação automatizado ao máximo.

2.2 ESPECÍFICOS

Como objetivos específicos do trabalho em questão pode-se citar os seguintes:

- Estudar as distribuição presentes e definir qual melhor se adapta ao público-alvo;
- Peneirar as distribuições de forma a oferecer o melhor para o usuário;

- Oferecer uma solução GNU / Linux que dispense custos monetários;
- Oferecer uma solução que esteja, se possível, 100% em português do Brasil;
- Oferecer uma solução que possa ser armazenada em um único CD-ROM de 700MB;
- Oferecer uma solução que não necessite de equipamentos de última geração para funcionar adequadamente.

3 Características de uma distribuição

A criação de uma distribuição GNU / Linux a partir do zero, isto é, sem tomar uma distribuição existente como base, é uma tarefa extremamente árdua. São diversos os motivos que dificultam tal tarefa, como a quantidade de bibliotecas e programas distintos com os quais se deve lidar – o que acaba causando a perda do foco na criação da distribuição em si. Além disso devemos considerar o fato de que uma boa parte do trabalho que será desenvolvido já foi, com suas devidas restrições, realizado na criação de outras distribuições GNU / Linux.

Levando em consideração tais fatores, decidiu-se embasar o trabalho em uma distribuição previamente existente e que já possuísse algumas das características desejadas. De maneira bem sucinta, é possível afirmar que o trabalho resume-se a modificar a distribuição tomada como base para atender às demandas propostas. Esta modificação engloba principalmente a inclusão e modificação de aplicações, documentação de ajuda e processo de instalação. É interessante manter os pacotes originais intocados, realizando as alterações através da inclusão de novos pacotes. Desta maneira é possível usufruir das atualizações de segurança oferecidos pela distribuição base.

Para realizar a seleção das inúmeras distribuições GNU / Linux existentes foram criados alguns critérios de seleção, que têm por intento não apenas facilitar a tarefa de seleção, mas também torná-la mais objetiva.

Embora o processo de customização seja baseado na alteração de partes que não estejam de acordo com o esperado, a escolha da distribuição base representa um importante passo. Alguns fatores não podem ser simplesmente modificados, e por isso é essencial que a distribuição escolhida esteja conforme determinadas características básicas, que serão vistas a seguir com respectivas explanações.

3.1 SOFTWARE LIVRE

Um dos maiores motivos para o sucesso do GNU / Linux está no fato deste tratar-se de software livre. A expressão “software livre” começou a ser usada

pela GNU para representar liberdade no mundo digital, e há uma diferença entre software livre e software grátis. Software livre não é o nome dado ao software que possa ser adquirido sem custo monetário – embora essa seja uma conseqüência na grande maioria dos casos. Conforme visto em [6] um software é considerado livre quando atende às quatro premissas básicas seguintes:

- liberdade de execução do programa;
- liberdade de estudar como o programa funciona e poder modificá-lo – esta liberdade implica no acesso ao código fonte;
- liberdade de redistribuir cópias deste software;
- liberdade de aperfeiçoar o software e redistribuir seus aperfeiçoamentos.

É importante diferenciar software livre de “open source” (código aberto). Embora as definições e as recomendações práticas sejam muito parecidas, os ideais e as questões éticas dos grupos que as criaram – Free Software Movement e Open Source Movement, respectivamente – são bastante distintas.

As distribuições GNU / Linux que incluem software proprietário foram descartadas do processo de escolha. Da mesma forma foram excluídas as distribuições que não podem ser obtidas de forma totalmente gratuita.

O uso de software livre e de distribuições que não acarretem gastos financeiros justifica-se não apenas por tratar-se de um trabalho acadêmico, mas também para lograr um produto com o menor custo possível.

3.2 LINUX STANDARD BASE

Nos últimos anos houve uma explosão no número de distribuições GNU / Linux disponíveis. Este crescimento caótico trouxe vários benefícios, sendo o principal a disseminação do GNU / Linux como sistema operacional nos mais variados ambientes. Apesar de promover a popularização, este crescimento também trouxe vários problemas.

Talvez o principal incômodo causado pela inúmera quantidade de diferentes distribuições seja a incompatibilidade entre elas. Em termos práticos isto significa que um determinado programa pode executar normalmente em uma distribuição, mas pode apresentar problemas em outra. Isto ocorre pois duas

distribuições diferentes podem conter bibliotecas / componentes diferentes; mesmo nos casos em que as bibliotecas / componentes necessários para a execução de uma aplicação estejam presentes, é possível que estas possuam incompatibilidades com a aplicação ou que se encontrem em localizações distintas. Um exemplo prático que ocorre é o plugin Flash Player (utilizado para visualização de páginas web) compilado com o GNU GCC 2.95 que não funciona com o navegador Mozilla compilado com o GNU GCC 3.2 ou superior. Neste caso o empecilho foi a incompatibilidade binária entre os compiladores de diferentes versões, que geram interfaces binárias incompatíveis.

Para minimizar os problemas de incompatibilidade, foi criado o Linux Standard Base Project, ou LSB. O LSB é um conjunto de definições e padrões (localização de bibliotecas, versões a serem utilizadas etc) que criam uma interface única e permitem que uma aplicação rode em qualquer distribuição conformante a estas regras. O LSB não se resume somente ao GNU / Linux e sim a qualquer sistema do tipo UNIX (UNIX-like). Embora o LSB seja um projeto aberto, dando liberdade para que qualquer um contribua, está sobre a tutela do Free Standard Group, uma organização independente que tem como objetivo acelerar o uso de software livre através do desenvolvimento de padrões para estes.

Após a criação do LSB as principais distribuições Linux vêm tentando se enquadrar nas normas definidas, no intuito de criar a desejada independência de distribuição. Mesmo assim algumas distribuições mais conservadoras decidiram manter os próprios padrões e não se adequar ao LSB, por motivos que estão fora do escopo e não serão aqui comentados.

Ser compatível com as normas do LSB não é um pré-requisito para a escolha da distribuição base, mas com certeza é uma característica bem-vista se presente na distribuição escolhida.

3.3 GERENCIADOR DE PACOTES

Já vimos que o Linux em si, e aqui entende-se como kernel, não possui muita aplicabilidade em um desktop se instalado isoladamente. Por ser um Sistema Operacional ele provê serviços que serão utilizados pelos programas, e não possui muita serventia se estiver desprovido destes.

No princípio esses programas eram distribuídos da mesma forma que o próprio kernel, apenas como código fonte. Era preciso então que o usuário realizasse uma série de tarefas para vê-lo funcionando: fazer o download da respectiva página, configurá-lo para sua máquina – aqui entra um dos problemas que o LSB tenta solucionar –, compilá-lo conforme instruções do desenvolvedor e instalá-lo.

Embora essa rotina funcione, ela tem algumas características altamente indesejáveis, como complexidade na instalação de um simples programa, facilidade de cometer erros ao ter que alterar configurações durante a instalação e quantidade infindável de repetições, uma vez que pode ser preciso instalar dezenas ou centenas de programas na mesma máquina. E, além do usuário, o desenvolvedor também sofria com essa metodologia arcaica.

Com a evolução no processo de instalação de contratempos problemas de mais alto nível começaram a surgir. O mais simples talvez fosse arranjar uma maneira de ter conhecimento de tudo o que havia instalado em seu computador; mas não era o único. Programas que precisavam de outros programas para rodar também apresentavam dificuldades na instalação, pois checar dependências é uma tarefa cansativa e ineficiente. E por fim podemos citar também os conflitos entre programas que não podem coexistir na mesma máquina.

Para tentar solucionar todos esses problemas – ou a grande maioria – foi introduzido o conceito de pacote (package), que pára de tratar um programa como um objeto isolado, adicionando conceitos como dependências e conflitos. Na prática um pacote é um arquivo que contém, além dos códigos binários, metadados que descrevem o conteúdo dele próprio. Dessa forma um pacote pode prover informações do seu conteúdo e suas relações com outros pacotes – as dependências e os conflitos.

Para efetuar a tarefa de controle dos pacotes foi introduzido o conceito de Sistema Gerenciador de Pacotes (Package Management System), que se resume a um sistema que mantém informações sobre pacotes instalados e pode instalar novos pacotes, realizar atualizações e remover pacotes já existentes. Os primeiros Sistemas Gerenciadores de Pacotes exerciam as funções básicas de verificação das relações entre os pacotes, o que facilitou muito a instalação e a manutenção de programas.

Desde a criação dos Sistemas Gerenciadores de Pacotes muitas

mudanças foram feitas em seus conceitos, uma delas foi a adição de pacotes virtuais, que propiciam um método para indicar que um pacote fornece alguma interface ou funcionalidade. Isto é feito introduzindo uma relação de fornecimento. Por exemplo, o pacote “mutt” requer um “agente transportador de correio”, e vários pacotes podem satisfazer essa condição, entre eles o “sendmail” e o “smail”. Então esses dois pacotes possuem em sua relação de fornecimento “agente transportador de correio”, e é essa relação que será verificada quando o pacotes “mutt” for instalado.

É importante salientar que o gerenciamento de pacotes é um dos principais fatores observados na escolha de uma distribuição base, e por isso em seguida é apresentada uma breve exposição sobre os principais e mais populares Sistemas Gerenciadores de Pacotes utilizados atualmente nas distribuições GNU / Linux.

3.3.1 Debian Package Manager

Debian foi uma das primeiras distribuições GNU / Linux a surgir, criada em 1993 por Ian Murdock, um estudante universitário vidrado em computadores. Sua motivação foi a decepção ao usar uma distribuição antiga que apresentava muitos problemas, vários dos quais não eram corrigidos em versões mais recentes. A Debian surgiu para o público em janeiro de 1994 e imediatamente chamou a atenção pelo seu Sistema Gerenciador de Pacotes intitulado dpkg. Pouco tempo depois a tarefa de desenvolvimento do dpkg foi delegada à Ian Jackson, que o transformou no que é hoje. Atualmente a Debian anuncia o dpkg como um sistema único e superior tecnicamente em relação aos usados pelas concorrentes.

Segundo [16]:

> Um pacote no formato Debian é composto por um único arquivo com extensão .deb que identifica esse como um pacote binário. Esse pacote contém todos os dados e arquivos executáveis necessários pelo pacote, assim como todas as informações de controle usadas pelo Gerenciador de Pacotes para instalar esses arquivos corretamente em seu sistema.

As Ferramentas Gerenciadoras de Pacotes da Debian acima citadas são três:

- *dpkg*: É a ferramenta que roda em linha de comando e acessa as

funções básicas do Gerenciador de Pacotes em um sistema Debian. Foi a primeira ferramenta a ser desenvolvida e provê acesso direto ao banco de dados de pacotes, assim como possibilidade de instalar, atualizar ou desinstalar pacotes individuais;

- *dselect:* É uma interface para o dpkg que facilita a busca e seleção de qualquer um dos pacotes que estejam presentes ou que se deseje instalar;
- *apt-get*: Foi criado para aumentar a funcionalidade do dpkg e propiciar um método melhor de instalação de vários pacotes do que o oferecido pelo dselect.

A clara vantagem do uso do Sistema Gerenciador de Pacotes presente no Debian Linux é o enorme número de pacotes .deb existentes e a grande facilidade encontrada na hora de atualizar pacotes ou até mesmo todo sistema, pois as avaliações de dependências e as escolhas de pacotes para satisfazer essas dependências são feitas de forma automatizada.

Como desvantagem pode-se citar o pouco uso deste sistema por outras distribuições – aquelas que o fazem geralmente são baseadas no próprio Debian.

3.3.2 Gentoo Package Manager

Gentoo é uma recente distribuição criada por Daniel Robbins e sua principal característica é poder transferir 100% do controle do sistema para as mãos do usuário. Através de seu Sistema Gerenciador de Pacotes, chamado Portage, é possível definir todas as variáveis desejadas no sistema e compilar todo o código fonte necessário para fazê-lo funcionar. Por essa alta taxa de customização e adaptabilidade o Gentoo é também chamado de metadistribuição.

Citando [9]:

> Num primeiro momento a idéia por trás do Portage parece muito similar com o tradicional sistema "ports" do BSD (Berkley Software Distribution). Ambos compilam diretamente do código-fonte e permitem que usuários instalem e desinstalem seguramente os programas do sistema, e ambos avaliam as dependências. Muitas idéias por trás do Portage são inspiradas no sistema "ports" do BSD, mas o Portage não é definitivamente mais uma cópia do "ports".

O Portage é o coração do Gentoo e trabalha com pacotes específicos, os

.ebuild. Conforme [4]:

> Scripts ebuild são a base para todo o sistema Portage. Eles contêm todas as informações necessárias para download, descompactação, compilação e instalação de um conjunto de códigos-fonte, assim como podem realizar uma pré(pós)-(des)instalação ou passos de uma configuração.

Embora o sistema Portage seja escrito em Python, os pacotes .ebuilds são escritos em bash, reproduzindo os comandos que seriam dados manualmente para efetuar a instalação do pacote.

Como ponto positivo no uso do sistema Portage podemos citar a alta customização e automaticidade do processo, além da facilidade de se realizar a atualização de todo o sistema.

Já para pontos negativos temos a conseqüência da alta customização, que é a complexidade ou excesso de opções para execução de uma tarefa – ou até a falta de uma interface gráfica correspondente. Enquanto um usuário avançado possa se beneficiar desse recurso, um leigo pode facilmente se assustar com a quantidade de parâmetros a serem escolhidos e desistir da tarefa.

3.3.3 Red Hat Package Manager

O Red Hat Linux surgiu como uma distribuição independente em 1994, sob responsabilidade de Marc Ewing. Em 1995 foi comprado por Bob Young e unido à sua companhia ACC Corporation, formando a Red Hat Software. Neste mesmo ano foi lançado o Red Hat Linux 2.0, com o novo Sistema Gerenciador de Pacotes chamado Red Hat Package Manager (RPM).

Muito do sucesso, crescimento e popularização do Red Hat Linux é atribuído às facilidades trazidas pelo RPM, que revolucionou o gerenciamento de pacotes. Como um software livre, a Red Hat Software incentivou o uso do RPM pelas mais diversas companhias, e a história do RPM evoluiu independente à da distribuição que o lançou. É possível ver que, embora hoje em dia o Red Hat Linux não exista mais sob esse nome, o RPM é o Sistema Gerenciador de Pacotes mais usado por distribuições GNU / Linux e o formato mais popular entre os criadores de aplicativos, além de já ter sido portado para outras plataformas, como o AIX, da IBM, e o IRIX, da Silicon Graphics.

A primeira versão do RPM já contava com inúmeras vantagens, como a

gestão automática dos arquivos de configuração, a simplificação do manuseio de vários pacotes e a facilidade no gerenciamento de pacotes. Mas inúmeras falhas haviam de ser corrigidas, e aqui podemos citar a falta de suporte para múltiplas arquiteturas, a falta de expansibilidade do formato de arquivos e os efeitos trazidos pelo código estar escrito em Perl: um desempenho longe do ideal e um tamanho exagerado.

Para tentar acabar com os problemas do RPM, seus criadores, Marc Ewing e Erik Troan, decidiram reescrevê-lo completamente, desta vez em C – melhor performance, menor tamanho –, repensar o formato do banco de dados usado – no intuito de melhorar o tempo de busca por dados e aumentar a confiabilidade – e criar uma biblioteca (rpmlib) com os procedimentos utilizados pelo RPM, para que outros programas também pudessem utilizá-los. Esse esforço resultou na base do RPM que é usado atualmente.

Um pacote .rpm é divido em quatro partes:

- *lead*: servia para identificação do pacote e deixou de ser usada devido à sua inflexibilidade. Foi substituída por uma estrutura *header*, solução encontrada para facilmente manipular dados através de um método padrão;
- *signature* (assinatura): estrutura do tipo *header* através da qual se pode garantir integridade e autenticidade do pacote .rpm;
- *header* (cabeçalho): estrutura do tipo *header* contendo todas as informações sobre o pacote, como nome, versão e lista de arquivos;
- *archive* (arquivo): contém os arquivos que compõem o pacote, compactados através do GNU zip.

O RPM se destaca dos outros Sistemas Gerenciadores de Pacotes por sua popularidade. Como se isso não bastasse, outro destaque do RPM é o de ser o Sistema Gerenciador de Pacotes escolhido como padrão do LSB.

As principais desvantagens do RPM residem na hora de atualizar uma distribuição baseada em RPM, na integração de sistemas com RPM de versões diferentes e na criação de um pacote RPM – devido a problemas que o LSB visa solucionar.

3.4 PROGRAMA DE INSTALAÇÃO

A instalação, uma tarefa imprescindível e de grande importância, é a principal responsável por afugentar usuários inseguros na hora de trocar de Sistema Operacional. Quando se fala de GNU / Linux o termo “instalação” pode ter inúmeras faces.

Desde métodos prosaicos que exigem que todos os hardwares sejam selecionados via uma interface de texto até instaladores quase totalmente automáticos com autodetecção de hardware e interface gráfica, a instalação de um sistema GNU / Linux pode variar imensamente de distribuição para distribuição. Podemos ilustrar essas diferenças com alguns exemplos práticos.

O SUSE Linux conta com o instalador YAST, que possui excelente autodetecção de hardware; todas as opções são definidas através de uma interface gráfica com a disponibilidade de assistentes e de um menu de ajuda com informações em diversas línguas. É um dos maiores exemplos em se tratando de amigabilidade e facilidades para o usuário. Outras distribuições que seguem essa linha na instalação são Red Hat Linux / Fedora Core e Mandrake Linux.

Do lado oposto o exemplo mais evidente é o do Debian Linux, que requer uma instalação feita totalmente através de uma interface texto, exigindo que o usuário selecione os drivers para os dispositivos presentes e necessitando que ajustes avançados sejam feitos para o completo funcionamento do sistema. Nessa linha podemos citar também o Slackware Linux, umas das primeiras distribuições a surgir, que prima pela estabilidade.

Diferente dos exemplos anteriores temos o Knoppix, distribuição GNU / Linux capaz de rodar através de um CD, sem necessidade de instalação alguma, que conta com autodetecção do hardware durante o processo de boot e criação de um disco virtual temporário para armazenamento de arquivos.

Outro exemplo distinto é encontrado nas chamadas distribuições-fonte, cujo representante mais evidente é o Gentoo Linux, que permite que o usuário faça o download do código fonte necessário – podendo selecionar até o compilador que será usado, assim como suas otimizações – durante o processo de instalação e obrigando-se assim a passar pela fase de compilação desse código – o que pode demorar várias horas ou até mesmo dias.

3.5 DOCUMENTAÇÃO E ASSISTÊNCIA

Como o GNU / Linux não é um produto de uma empresa e é distribuído livremente, não há um manual oficial que o acompanhe, tão pouco assistência técnica por parte do fabricante. Sendo assim, a documentação disponibilizada é indispensável, pois é a fonte de onde as perguntas serão respondidas.

A documentação é feita do mesmo modo que seu código: através de parceria entre diversos programadores e entidades ligadas ao mundo do software livre. Há o Projeto de Documentação Linux (Linux Documentation Project – LDP) cujo objetivo é servir de base de dados para quem deseja procurar qualquer tipo de informação – manuais, informações gerais, HOWTOs e mais – sobre sistemas GNU / Linux. Toda a documentação existente no LDP é aberta, pode ser copiada e lida por quem desejar. Essa documentação abrange todo e qualquer tipo de procedimento usual realizado em um sistema, começando na instalação de um aplicativo qualquer, passando por uma configuração de um modem e terminando em um método para compartilhar acesso à Internet.

Mesmo com todas essas formas de documentação, cada distribuição possui peculiaridades e atributos próprios. Logo, informações ou procedimentos genéricos podem não convergir para o mesmo resultado em distribuições distintas. Este é um dos grandes motivos pelo sucesso de uma ou outra distribuição GNU / Linux: a facilidade que o usuário tem em encontrar informações que o ajudem em determinada ocasião. Por mais experiente que ele seja, é indispensável que haja um lugar onde buscar auxílio quando não consegue realizar determinada tarefa. Para um usuário leigo isso se torna ainda mais importante – sob certo ponto de vista, até essencial.

Essas informações específicas para cada distribuição vão desde tabelas de compatibilidade de hardwares (é importante saber se o hardware que você possui irá funcionar de forma correta) até roteiros com ajuda na hora da instalação (um dos processos mais destoantes de distribuição para distribuição) e fazem parte do grupo de características que contam na avaliação de uma distribuição.

Aqui podemos ver a importância de a distribuição pretendida possuir suporte para a língua local: de pouco adianta informações abundantes e claras se o usuário não conseguir compreendê-las.

4 Distribuição escolhida

Levando-se em consideração os fatores citados no capítulo 3 e as necessidade impostas, a distribuição escolhida como base para a realização do trabalho foi o Fedora Core.

O Fedora Project, ao lado do Red Hat Enterprise Linux, é um dos descendentes do extinto Red Hat Linux e possui esse nome por tratar-se de um projeto, e não de uma distribuição final. É um projeto aberto que tem como objetivo trabalhar lado-a-lado com a comunidade Linux a fim de construir um Sistema Operacional de uso genérico construído exclusivamente em cima de software livre. O desenvolvimento do Fedora Project é feito em um fórum público e versões periódicas do Fedora Core (a distribuição em si) são lançadas. Embora seja aberto e utilize somente software livre a equipe de engenheiros da Red Hat participa do Fedora Project, porém sem suporte de qualquer tipo proveniente da Red Hat Software; apenas fóruns e grupos de discussões são disponibilizados para esclarecimentos e ajuda aos usuários.

O primeiro e principal motivo pelo qual o Fedora Core foi escolhido como distribuição base é a política adotada pelo Fedora Project, que tem objetivos bem definidos e condizentes com aqueles definidos para este trabalho. O mais importante deles talvez seja o de utilizar-se somente de software livre, o que vem perfeitamente ao encontro de uma das características necessárias para a distribuição base.

Outro objetivo colaborante com o trabalho descreve a priorização da usabilidade e da filosofia de “apenas funcionar” (descrito em [15] como “just works” philosophy), utilizando-se sempre das configurações e características padrões e livrando o usuário de todos incômodos possíveis.

Além dos citados, outros objetivos pertinentes do Fedora Project são:

- método fácil de fazer atualizações, com mínimas modificações em arquivos de configuração;
- adoção dos softwares mais populares e mais conhecidos, livrando o usuário de programas desconhecidos e criando pacotes padrões de softwares;
- promoção de perspectivas globais, com suporte a tantas línguas e localidades geográficas quantas forem possíveis, tornando o

produto universal.

Todas essas características ajudaram muito na escolha do Fedora Core como distribuição base, e economizaram muito do trabalho que seria feito. Escolhendo o Fedora Core como base não é necessário remover nenhum pacote essencial à distribuição que seja proprietário (o que ocorre com outras distribuições, por exemplo). Por tratar-se de uma distribuição destinada ao usuário final, muitas tarefas são automatizadas, e os menus e atalhos já estão organizados de uma forma satisfatória para um usuário leigo.

Embora o Fedora Core não possua o certificado LSB, o Red Hat Linux 9 – última versão lançada sob esse nome – possui certificação LSB 1.3. Tendo em vista que a Red Hat é uma das empresas que apóiam o LSB e que o Fedora Core é um descendente, podemos afirmar que não se trata de uma distribuição destoante deste padrão.

Naturalmente, o Fedora Core utiliza como Sistema Gerenciador de Pacotes o RPM, que foi comentado detalhadamente no capítulo 3.3.3. O programa utilizado para verificar a disponibilidade e realizar as atualizações do sistema é o up2date. Um dos principais atrativos do up2date é a sua capacidade de superar uma das deficiências do sistema gerenciador de pacotes RPM, que é a auto-resolução de dependências.

O programa de instalação utilizado pelo Fedora Core é o Anaconda, o mesmo utilizado pelo antigo Red Hat Linux. Fácil de usar e com uma interface gráfica amigável, o Anaconda possui um menu de ajuda que pode estar em várias línguas disponíveis, e exibe explicações detalhadas sobre cada passo que deve ser dado durante a instalação. Ainda assim, o Anaconda requisita decisões comuns durante a instalação de uma distribuição GNU / Linux que podem confundir pessoas menos preparadas, como tipo do sistema de arquivos a ser utilizado e informações sobre o particionamento do disco.

5 Customização da distribuição

O principal objetivo da customização é tornar a distribuição o mais amigável possível. O ideal é facilitar cada tarefa para o usuário sem que ocorra perda de flexibilidade, mas nem sempre isso é possível. Reduzir o número de tomadas de decisão muitas vezes simplifica a tarefa, mas por outro lado torna-a menos flexível. Este é o caso de grande parte das customizações realizadas neste trabalho.

Reduzir o número de aplicações para que a distribuição inteira possa ser armazenada em um único CD-ROM também traz conseqüências que afetam a diretamente a versatilidade da distribuição. Muitas aplicações que possivelmente interessariam a um usuário um pouco mais avançado foram eliminadas para reduzir o espaço de armazenamento.

O nome escolhido para a nova distribuição foi “Winona”, que na linguagem Dakota significa “primogênito”.

Embora um dos objetivos seja tornar a distribuição o menos exigente possível em termos de hardware para poder abranger um maior número de máquinas, é necessário no mínimo um computador com um processador Pentium ou superior, 64MB de memória RAM, 2GB de disco rígido e um drive de CD-ROM para o bom desempenho da Winona. Esses requisitos são exigidos principalmente pelo Metacity, o Gerenciador de Janelas do GNOME e pela suíte de aplicativos para escritório OpenOffice.

5.1 ANACONDA

Geralmente o programa instalador reconhece e configura o hardware, cria o sistema de arquivos e copia as aplicações e os componentes necessários para executar o sistema operacional. O programa de instalação do Fedora Core é conhecido como Anaconda e a versão utilizada neste trabalho é a 9.2, que acompanha a versão 1.0 do Fedora Core.

O Anaconda pode executar em modo gráfico, com a interação do usuário, ou através de scripts em modo não-atendido, que pode dispensar completamente a

interação com o usuário. Este último modo, conhecido como “kickstart mode”, será utilizado como artifício para reduzir a intervenção do usuário durante o processo de instalação, cumprindo o objetivo de privar o usuário de tomadas de decisões excessivas

A customização do instalador Anaconda não se limita somente à aplicação do modo não-atendido, também é preciso personalizar todas as telas, modificar a estrutura de arquivos, adicionar pacotes criados, pacotes de atualizações e pacotes extras, remover os pacotes excedentes e gravar o CD personalizado.

5.1.1 Kickstart

O método kickstart existe para suprir uma necessidade que muitos administradores de sistema possuem, a instalação automática. Após realizada uma instalação típica, com a intervenção do usuário, o Anaconda cria automaticamente um arquivo contendo as respostas para todas as perguntas feitas durante a instalação. Este arquivo, que encontra-se no diretório “/root”, é chamado anaconda-ks.cfg e pode então ser utilizado para fazer a instalação automática nos demais equipamentos.

Como já foi destacado anteriormente, certas decisões não serão tomadas pelo usuário, e uma delas é o particionamento do disco rígido. Como esta distribuição não se propõe a viver em harmonia com outros Sistemas Operacionais, todo o conteúdo do disco é apagado e o Anaconda decide automaticamente como particioná-lo. São criadas três partições, uma de boot, uma para dados e outra para swap. O sistema de arquivos utilizado na partição de dados e de boot é o ext3. A partição de boot ocupa entre 75 MB e 100 MB, dependendo do tamanho do disco rígido. A partição de swap geralmente é duas vezes maior que a quantidade de memória RAM do sistema.

A resolução do vídeo é definida para 800 x 600, já que a distribuição se propõe a executar até mesmo nos equipamentos mais modestos. A senha do usuário root é definida como “fedora”, mas como será visto mais adiante, essa distribuição não se propõe a operar de modo multi-usuário. A linguagem do sistema é definida para português do Brasil. O dispositivo de rede é configurado para receber um endereço IP via DHCP. O fuso-horário é definido arbitrariamente para o de Brasília.

A única decisão que cabe ao usuário durante todo o processo de

instalação é definir qual o mapa do teclado a ser utilizado, já que não existe maneira alguma de fazer a detecção automática dos diferentes modelos de teclados – levando em consideração que a detecção teria que abranger todas as tecnologias atualmente existentes.

O arquivo ks.cfg, mostrado no anexo 8.3, deve estar na raiz do CD-ROM.

5.1.2 Comps.xml

O arquivo comps.xml define quais são os grupos de instalação, assim como os pacotes pertencentes a cada grupo. Numa instalação típica o usuário pode escolher quais grupos ou quais pacotes individuais devem ser instalados. Como exemplo existe um grupo para o Gnome chamado gnome-desktop, que engloba todos os pacotes necessários para o bom funcionamento do Gnome. Em uma instalação não-atendida, os grupos que devem ser instalados são definidos no item %packages do arquivo ks.cfg.

É importante observar que, assim como existem dependências entre aplicações, também existem dependências entre os grupos. Por exemplo, o grupo gnome-desktop não pode ser instalado na ausência do grupo base-x, que contém a base do X Window System. As dependências, tanto entre grupos como entre pacotes individuais, são calculadas e satisfeitas automaticamente durante a instalação.

Na última versão do Red Hat Linux – e posteriormente no Fedora Core – o arquivo comps.xml progrediu de um simples arquivo texto com algumas tags para – como o próprio nome sugere – um arquivo .xml. Essa evolução facilitou muito a definição de grupos e pacotes neste arquivo e sua respectiva leitura pelo instalador Anaconda. Como foram criados novos pacotes para a distribuição Winona, foi preciso alterar o arquivo comps.xml a fim de que eles fossem incluídos na instalação padrão.

A customização do arquivo comps.xml consiste em adicionar os pacotes criados especialmente para a distribuição customizada nos seus grupos correspondentes. Além disso, é preciso remover todos os pacotes presentes na distribuição base, porém ausentes na distribuição customizada.

Esta parte do trabalho é a que mais despende tempo, já que o Fedora possui aproximadamente 1600 pacotes, armazenados em três CDs, e a proposta desta distribuição é criar um produto compacto, que pode ser armazenado em

apenas um CD.

O método utilizado para formular a lista dos pacotes da distribuição foi listar todos os pacotes da instalação padrão do Fedora, removendo aqueles que fossem classificados como desnecessários. O critério de remoção de pacotes é tratado mais adiante. Para gerar a lista, foi criado o programa do anexo 8.5.

5.1.3 Personalização do CD

A personalização do CD consiste basicamente em obter uma cópia do CD de instalação do Fedora, copiar todo o seu conteúdo para o disco rígido, adicionar os pacotes criados, substituir os pacotes corrigidos pela equipe de manutenção da distribuição base, modificar os arquivos de configuração e então gravar o CD.

O passo mais simples é alterar o nome “Fedora” para “Winona”, no diretório raiz do CD. Também é preciso alterar o arquivo /isolinux/isolinux.cfg (que pode ser visto no anexo 8.4) para que o processo de instalação comece prontamente, utilizando o método kickstart, uma vez que o boot do CD original do Fedora Core apresenta inicialmente uma tela na qual o usuário decide que tipo de instalação será realizada.

O passo mais trabalhoso e que mais consome tempo na personalização do CD é a eliminação e adição de pacotes. Muitas vezes este processo é descrito como terrível por lidar com as dependências entre os pacotes. Incluir um pacote pode implicar em incluir vários outros, já que é preciso satisfazer suas dependências. Da mesma maneira, excluir um pacote pode implicar em excluir vários outros, já que certos pacotes podem estar presentes apenas para satisfazer as dependências de outros. Para dificultar ainda mais, a substituição de pacotes já presentes por suas versões atualizadas também apresenta os problemas acima citados.

5.2 PERSONALIZAÇÃO DE PACOTES

Como dito anteriormente o Fedora Core é baseado no Sistema Gerenciador de Pacotes RPM. Isso significa que para inserir um novo pacote ou modificar um pacote existente na distribuição é necessário trabalhar com o RPM.

5.2.1 Criação de um .rpm

A criação de um pacote .rpm é relativamente simples para aqueles que estão acostumados a compilar programas diretamente do código-fonte. São necessários os seguintes passos para criar um pacote .rpm:

- Obter o código-fonte que você irá transformar em .rpm;
- Criar um “patch” de todas as alterações necessárias para este código-fonte ser compilado no sistema desejado;
- Criar um arquivo .spec para o pacote;
- Garantir que os arquivos estejam nos lugares apropriados;
- Gerar o pacote .rpm.

Em uma situação normal será gerado tanto o .rpm binário quanto o .rpm com o código-fonte.

O arquivo .spec mencionado acima é imprescindível para a criação de um pacote .rpm. Ele contém as informações de como o pacote deve ser construído assim como a lista de todos arquivos binários que serão instalados. No cabeçalho de um arquivo .spec está uma simples descrição do pacote, nome, versão, release, copyright, grupo a qual o pacote pertence, endereço onde o código-fonte pode ser encontrado, diretório onde será instalado e uma descrição mais longa que permite várias linhas de explicação a respeito do pacote .rpm. Após o cabeçalho o arquivo .spec contém os campos “prep” (roteiro para descompactação do código-fonte, aplicação de patchs e modificações deixando-o pronto para ser compilado), “build” (compilação do código), “install” (instalação do código no sistema) e “clean” (para total remoção do que foi instalado). Ainda existem campos opcionais que permitem que sejam programadas alterações pré/pós instalação/desinstalação. Completando o arquivo .spec temos o campo “files” que lista os arquivos gerados que serão instalados no sistema e o campo “changelog”, que contém informações sobre modificações feitas no .spec.

Alguns arquivos .spec utilizados encontram-se por completo no capítulo 8.5 dos anexos.

Antes de começar a gerar os pacotes .rpm é necessário montar uma árvore de criação (build tree), que consiste em uma hierarquia de cinco diretórios com finalidades específicas:

- BUILD: diretório onde realmente ocorre a criação de um .rpm;
- SOURCES: é o diretório onde os pacotes com o código fonte (e patches) devem estar;
- SPEC: local dos arquivos .spec;
- RPMS: diretório destino dos .rpm binários;
- SRPMS: diretório destino dos .rpm com código-fonte.

Com essa estrutura de diretórios é possível gerar e fazer testes com arquivos .rpm. Através destes testes o arquivo .spec vai sofrendo incrementações até que o .rpm resultante esteja de acordo com o esperado.

5.2.2 OpenOffice

A instalação do OpenOffice.org do Fedora Core possui graves problemas com relação aos objetivos desse trabalho. As interfaces de usuários dos programas, assim como o corretor ortográfico, foram traduzidos para o português de Portugal. Não foi incluído suporte para hifenação e a ajuda não foi traduzida para nenhuma variante do português, permanecendo totalmente em inglês.

A melhor maneira de solucionar estes inconvenientes seria recompilar o OpenOffice.org, adicionando o suporte a língua nativa brasileira, já que o código fonte inclui suporte para esta língua. Apesar de simples, esta solução provou ser impraticável, já que a compilação do OpenOffice.org mostrou-se impossível no Fedora Core, por problemas com a máquina virtual java.

No sítio openoffice.org.br é possível encontrar uma versão compilada do OpenOffice.org em português do Brasil. Esta versão não possui hifenação e nem sequer ajuda em português. Entretanto, o principal problema é a incompatibilidade entre esta versão e a instalada pelo Fedora Core, que possui vários itens personalizados, como a localização dos arquivos executáveis, de seus atalhos, além da associação dos tipos de arquivos etc.

Também no openoffice.org.br encontram-se um “dicionário” de hifenação e o projeto de tradução da ajuda para o português do Brasil, que é recente mas já encontra-se em um estado bastante avançado; a única porção ainda não totalmente traduzida é o índice.

Adaptar a versão pré-compilada do openoffice.org.br seria uma tarefa muito trabalhosa, portanto decidiu-se manter a versão original do Fedora. A solução

encontrada foi criar um pacote adicional contendo apenas os arquivos responsáveis pela tradução. A criação de um novo pacote implica na alteração do processo de instalação.

Seguindo os padrões de nomenclatura do Fedora Project, o novo pacote foi chamado de openoffice.org-i18n-pt_BR. Este novo pacote resulta da união de três componentes distintos: os arquivos de tradução da interface de usuário, a ajuda e os dicionários de hifenação e de correção ortográfica. Nenhum dos arquivos deste pacote possui dependência com arquivos externos ao openoffice.org, como, por exemplo, bibliotecas do sistema, o que possibilita que este pacote seja adicionado a qualquer instalação do openoffice.org.

5.2.3 Navegação na Internet

A experiência de navegar na Internet pode ser bastante frustrante para um usuário do Fedora Core, já que está distribuição não possui suporte para duas tecnologias atualmente bastante difundidas na Internet: Java 2, da Sun Microsystems, e Macromedia Flash.

Para eliminar estas limitações foram criados dois pacotes extras, um plugin para a máquina virtual Java e outro para o flash-player. Estes dois plugins funcionam apenas no Mozilla, que é o navegador web padrão do Fedora Core. O plugin Java possui um pré-requisito, que é a existência de uma máquina virtual Java, portanto foi necessário a criação de outro pacote. O outro navegador do Fedora Core, o Konqueror, não necessita de um plugin para habilitar o suporte a Java; a existência da máquina virtual é suficiente. Por outro lado, a combinação gcc 3.x, konqueror e flash-player mostrou-se não-factível, por problema de interfaces binárias incompatíveis.

Embora essas duas tecnologias tragam facilidades e avanços para a navegação na Internet, elas não estão incluídas na distribuição base por possuírem licenças não-conformantes com a livre redistribuição.

Um outro pequeno problema relacionado com a experiência de uso da Internet foi observado com o aplicativo Gaim, que é uma ferramenta de Mensagens Instantâneas de multi-protocolo, isto é, uma ferramenta capaz de se comunicar com diversas redes de mensagens como ICQ, MSN, Jabber, IRC etc. O item de menu do Gaim não havia sido traduzido para o português e a solução foi bastante simples, bastando fazer uma pequena alteração no código fonte do pacote.

5.2.4 Telas Personalizadas

A personalização das imagens de fundo e das telas em que são encontrados os logos do Fedora Core é uma tarefa bastante ordinária, porém fundamental na personalização da distribuição. Depois de descobrir onde estão armazenadas as imagens e a que pacotes elas pertencem, basta fazer a substituição na fonte do pacote.

O RPM possui uma função que informa a qual pacote um determinado arquivo pertence, portanto esta tarefa é bastante fácil de executar. Encontrar as imagens pode ser um pouco mais trabalhoso, mas a grande maioria delas pode ser vista no diretório /usr/share/pixmaps/ ou em algum de seus subdiretórios.

Os pacotes alterados foram: anaconda-images, que possui as imagens mostradas durante o processo de instalação, desktop-backgrounds, que possui as telas de fundo da área de trabalho do usuário, e gnome-session, que possui a tela que é mostrada enquanto o Gnome é carregado.

Além destes outros pacotes foram alterados para modificar atributos visuais presente no Fedora Core, entre eles redhat-artwork, com imagens presente no menu do Sistema, e redhat-menus, com as descrições presentes no menu do GNOME.

5.2.5 Autologin

Um das decisões mais difíceis no trabalho foi a adoção do autologin. Deixando de lado as questões de segurança, é bastante difícil avaliar até que ponto o autologin facilita a vida do usuário e até que ponto atrapalha.

A dificuldade encontra-se justamente em determinar quantos serão os usuários do computador. Quando utilizado somente por um usuário o autologin certamente é bem-vindo, mas quando este número aumenta os conflitos nas preferências pessoais geralmente não compensam a facilidade trazida.

Decidiu-se que a distribuição opera em modo "single-user", portanto o autologin foi adotado. Para tanto foi preciso criar um usuário diferente adicional, já que não é aconselhável nem permitido realizar autologin do root. Foi necessário fazer uma pequena alteração no arquivo de configuração do gdm, que é o gestor de início do Fedora Core, e também no script de pós-instalação do Anaconda.

5.2.6 Pacotes excluídos

Como explicado anteriormente, muitos pacotes que não são essenciais para o funcionamento do sistema ou para os objetivos traçados foram eliminados. Nos casos em que existem vários pacotes com funcionalidades similares, foi escolhido apenas um para permanecer na distribuição. Esta eliminação de pacotes desnecessários contribui para a diminuição do tamanho da distribuição e para simplificações como, por exemplo, a redução do número de itens no menu.

Como exemplos de pacotes que foram removidos temos aqueles relacionados ao Gerenciador de Janelas KDE (kdegames, kdelibs, kde-i18n-Brazil e demais) já que o GNOME será utilizado, pacotes de aplicativos raramente utilizados (XFree86-100dpi-fonts, mtr-gtk), pacotes geralmente utilizados por usuários mais avançados (emacs e pacotes relacionados, redhat-config-kickstart) e pacotes com aplicativos não-essenciais (xchat, por exemplo).

5.3 PROBLEMAS ENCONTRADOS

Boa parte dos problemas foi encontrada na customização do programa de instalação do Fedora Core, o Anaconda. Para modificar ou até mesmo utilizar de maneira eficiente um software complexo, como é o caso do Anaconda, é fundamental que exista algum tipo de material de apoio, como tutoriais ou artigos baseados em tarefas. Preferencialmente, estes documentos devem ser aprovados pelos desenvolvedores do software. Uma boa indicação de aprovação é a disponibilização dos documentos no site do software.

Apesar de já existir um projeto de documentação do Fedora Project, incluindo todos os seus componentes, a documentação tanto para usuários finais quanto para desenvolvedores praticamente não existe no site do Fedora Project. Em particular, existe uma disposição para criar um guia de customização do Anaconda; a inexistência desse guia dificultou muito o trabalho, já que o customização do Anaconda foi o ponto de partida.

Através de ferramentas de procura na Internet, foi possível encontrar alguns artigos escritos por desenvolvedores que também sentiram falta de documentação oficial. Estes artigos geralmente cobriam as tarefas mais simples, como gravar um CD com o Anaconda.

Como principal alternativa para a ausência de documentação oficial, existe uma lista de discussão para desenvolvedores da Fedora e de seus componentes. Esta lista de discussão foi bastante útil e provou ser suficiente para solucionar grande parte dos problemas encontrados. Por outro lado, a utilização de uma lista de discussão apresenta alguns problemas – como confiança na fonte da informação – que não serão tratados neste texto.

É importante salientar que o Anaconda sofreu mudanças estruturais consideráveis recentemente. Estas alterações tornaram inútil grande parte da documentação existente sobre o assunto, tanto na lista de discussão quanto nos artigos publicados. Nos poucos casos em que a documentação extra-oficial atualizada não estava disponível, uma análise do código fonte era necessário para solucionar os problemas.

Outro fator que dificultou ainda mais a obtenção de documentação foi a questão de que o Fedora Core está na sua primeira versão, e as informações referentes a ele ainda não estão maduras. Em contrapartida temos o fato do Fedora Core ser descendente direto do Red Hat 9, possibilitando que muita informação referente à este seja aplicado ao outro.

Na parte de modificação / inclusão de pacotes, as principais dificuldades ficaram por parte das incompatibilidades. Tanto novos aplicativos quanto novas versões de aplicativos existentes podem apresentar incompatibilidades que, ao tentarem ser resolvidas modificando-se o arquivo já existente, e não o novo, provocam incompatibilidades em cadeia, tornando o processo lento e altamente ineficiente.

6 Conclusão e trabalhos futuros

Apesar do esforço despendido, nem todos os objetivos traçados foram alcançados de forma plena. Não foi possível colocar todos os pacotes em apenas um CD. Também não foi possível livrar totalmente a distribuição de software proprietário.

Criar uma distribuição sem tomar outra como base provou ser ineficaz, pois a relação trabalho / benefício não é boa. O ideal é deixar para a distribuição base a inacabável tarefa de manter os pacotes livres de vulnerabilidades.

Como principal trabalho futuro podemos citar a manutenção da distribuição. A manutenção da distribuição engloba atualizações de pacotes, adição de novos pacotes que forem surgindo e até mesmo atualização da distribuição base, conforme novas versões desta forem lançadas.

Além disso é preciso criar uma opção para operar no modo sem mídia, para que até mesmo os equipamentos mais modestos possam executar aplicativos que requerem maiores recursos, como o OpenOffice.

Outro trabalho futuro de grande valia seria a criação de um LiveCD, que consiste em uma distribuição GNU / Linux capaz de rodar diretamente do CD, sem a necessidade de instalação. Embora esse fato traga alguns incômodos como o tempo necessário para a inicialização e a necessidade de reconhecimento de todo o hardware a cada vez que é dado o boot, é uma alternativa que facilita muito a vida do usuário novato, haja visto que um dos processo mais incômodos (a instalação do Sistema Operacional) não é essencial.

7 Bibliografia

[1] AKKERMAN, Wichert. **Linux Package Management.** 2000. Disponível em: <http://www.wiggy.net/presentations/>. Acesso em: dezembro de 2003.

[2] BAILEY, Edward C. **Maximum RPM**: Taking the Red Hat Package Manager to the limit. 2ª edição. Red Hat Inc., 2000.

[3] BARNES, Donnie. **RPM HOWTO**. 2002. Disponível em: <http://www.rpm.org/RPM-HOWTO/>. Acesso em: dezembro de 2003.

[4] DAVIES, Donny, et al. **Gentoo Linux Developers HOWTO**. 2003. Disponível em: <http://www.gentoo.org/doc/en/gentoo-howto.xml>. Acesso em: dezembro de 2003.

[5] FREE SOFTWARE FOUNDATION. **GNU General Public License**. Disponível em: <http://www.gnu.org/copyleft/gpl.html>. Acesso em: novembro de 2003.

[6] FREE SOFTWARE FOUNDATION. **The Free Software Definition**. Disponível em: <http://www.gnu.org/philosophy/free-sw.html>. Acesso em: dezembro de 2003.

[7] FREE STANDARD GROUP. **Linux Standard Base Specification 1.3**. Disponível em: <http://www.linuxbase.org/spec/oldspecs.shtml>. Acesso em: dezembro de 2003.

[8] LAWYER, David S. **Linux Documentation Project Manifesto**. Disponível em: <http://www.tldp.org/manifesto.html>. Acesso em: dezembro de 2003.

[9] LOCKE, Bruce A., et al. **Portage Manual**. Disponível em: <http://www.gentoo.org/doc/en/portage-manual.xml>. Acesso em: dezembro de 2003.

[10] LWM.net. **The LWN.net Linux Distribution List**. Disponível em: <http://old.lwn.net/Distributions/>. Acesso em: dezembro de 2003.

[11] OLIVEIRA, Rômulo Silva de; CARISSIMI, Alexandre da Silva; TOSCANI, Simão Sirineo. **Sistemas Operacionais.** 2ª edição. Porto Alegre: Sagra Luzzato, 2001.

[12] RED HAT INC. **Red Hat Installation Program**. Disponível em: <http://fedora.redhat.com/projects/anaconda-installer>. Acesso em: dezembro de 2003.

[13] RED HAT INC. **Red Hat Linux Customization Guide**. 2003. Disponível em: <http://www.redhat.com/docs/manuals/linux/RHL-9-Manual>. Acesso em: dezembro de 2003.

[14] RED HAT INC. **The Anaconda-devel-list Archives**. Disponível em: <https://listman.redhat.com/archives/anaconda-devel-list>. Acesso em: dezembro de 2003

[15] RED HAT INC. **The Fedora Project Objectives**. Disponível em: <http://fedora.redhat.com/about/objectives.html>. Acesso em: dezembro de 2003.

[16] SCHEETZ, Dale. **Dwarf's Guide to Debian GNU/Linux**. 2001. Disponível em: <http://www.debian.org/doc>. Acesso em: novembro de 2003.

[17] STALLMAN, Richard. **Linux and the GNU Project.** 2003. Disponível em: <http://www.gnu.org/gnu/linux-and-gnu.html>. Acesso em: dezembro de 2003

8 Anexos

8.1 ARTIGO

Estudo de Implementação de uma Distribuição GNU / Linux

Crineu Tres[1], Ulisses Muniz de Queiroz[2]

[1] crineu@inf.ufsc.br,
[2] muniz@inf.ufsc.br

Resumo

Este trabalho se propõe a criar uma distribuição GNU / Linux para ser usada em praticamente qualquer computador e que possa ser adquirida de forma gratuita, além de possui todas funcionalidades de um Sistema Operacional proprietário.
Uma distribuição base foi escolhida e então modificada de acordo com o desejado. Algumas das modificações mais importantes dizem respeito à diminuição do tamanho, disponibilização dos softwares na língua nativa e inclusão de aplicativos considerados convenientes.

Palavras-chave: Linux, Distribuições GNU / Linux

Abstract

The purpose of this work is to create a GNU / Linux distribution to be used in practically any computer and acquired at no financial cost, besides having all features of a proprietary Operational System.
A base distribution was chosen and modified to be according the project. Some of the major modifications are about size restrictions, native language software availability and inclusion of helpful software

Keywords: Linux, GNU / Linux Distributions

Introdução

Embora a criação do primeiro computador tenha ocorrido há mais de 50 anos, apenas nas últimas décadas do século XX que este pequeno aparelho conquistou a onipresença que possui atualmente e sua difusão como ferramenta de trabalho, estudo e entretenimento.

Um Sistema Operacional em especial esteve intrinsecamente ligado com o crescimento do próprio PC: trata-se do Sistema Operacional Windows, da Microsoft, que por muito tempo dominou praticamente sem concorrência o mercado dos desktops.

Esse cenário começou a mudar em meados da década de 90, quando o sistema GNU / Linux deixou de ser exclusividades de servidores e centros de pesquisa e invadiu os desktops. Embora inicialmente não fosse um sistema pronto para o usuário doméstico, ganhou espaço aos poucos e começou a conquistar os usuários mais experientes.

Desde essa época, a grande batalha está em convencer o usuário de que o GNU / Linux não é um sistema operacional que necessite de conhecimentos profundos para ser utilizado.

Distribuições GNU / Linux, sistema GNU e o kernel Linux geralmente são confundidos entre si, e em muitas ocasiões méritos e deméritos de um e de outro são concedidos ao elemento errado.

A história do Linux começa no início da década de 90, quando Linus Torvalds resolveu criar um Sistema Operacional melhor do que o Minix – sistema baseado no UNIX, porém

muito mais simples O trabalho de Torvalds evoluiu para tornar-se o que hoje conhecemos por Linux, o kernel da nossa distribuição.

O kernel é o núcleo do sistema operacional, isto é, o responsável pelas tarefas mais importantes realizadas em um computador, como gerência de memória, gerência de processos e manipulação do sistema de arquivos. Segundo [5] "A arquitetura de um sistema operacional corresponde à imagem que o usuário tem do sistema, a forma como ele percebe o sistema. Essa imagem é definida pela interface através da qual o usuário acessa os serviços do sistema operacional. Essa interface, assim como a imagem, é formada pelas chamadas de sistema e pelos programas de sistema."

As chamadas de sistema são implementadas pelo kernel, e é através delas que os programas de sistema se comunicam com o sistema operacional, requisitando serviços necessários à sua execução. O Linux é o responsável pela implementação e realização de todos os serviços necessários para habilitar a execução dos programas de sistema.

O sistema GNU – GNU's Not Unix (GNU não é UNIX) – tem seu início um pouco antes, em 1983, quando Richard Stallman decidiu criar um sistema operacional inteiramente novo e free, baseado no UNIX. Sua pretensão inicial era conceber um kernel e utilitários para criar e rodar programas em C: um editor, um shell, um compilador C, um ligador e um assembler.

No início dos anos 90 o projeto GNU estava quase pronto, com exceção de seu kernel. Porém nessa época o kernel Linux já era uma realidade e o que houve então foi o trabalho de união entre o kernel de Torvalds e o sistema de Stallman, resultando em um sistema operacional completo e free, o GNU / Linux.

Uma distribuição GNU / Linux, por sua vez, é o aglomerado de diversos aplicativos rodando sobre o Sistema Operacional GNU / Linux. Embora todas as distribuições GNU / Linux fundamentem-se no mesmo âmago, as diferenças essenciais encontram-se na documentação e nos aplicativos oferecidos por cada uma.

Certas características são fundamentais para o sucesso de uma distribuição, e tais características são definidas de acordo com o público alvo. Uma distribuição voltada para o uso doméstico por pessoas consideradas leigas – como a proposta por esse trabalho – deve zelar pela facilidade do uso, simplicidade, comodidade e uma boa aparência visual, de modo a agradar o usuário.

Uma distribuição GNU Linux em português do Brasil, de fácil instalação, com aplicações de Internet / escritório e com um menu de navegação simples é uma solução apropriada para adotar-se, por exemplo, em iniciativas de inclusão digital.

Existem várias distribuições com algumas das características citadas acima, mas nenhuma que possua todas. O objetivo principal deste trabalho é a criação de uma distribuição GNU / Linux ideal para primeiro contato entre principiantes e computadores.

Características de uma Distribuição

A criação de uma distribuição GNU / Linux a partir do zero, sem tomar uma distribuição existente como base, é uma tarefa extremamente árdua. Além disso devemos considerar o fato de que uma boa parte do trabalho que será desenvolvido já foi, com suas devidas restrições, realizado na criação de outras distribuições GNU / Linux.

Levando em consideração tais fatores, decidiu-se embasar o trabalho em uma distribuição previamente existente e que já possuísse algumas das características desejadas. De maneira bem sucinta, é possível afirmar que o trabalho resume-se a modificar a distribuição tomada como base para atender às demandas propostas. Esta modificação engloba principalmente a inclusão e modificação de aplicações, documentação de ajuda e processo de instalação. É interessante manter os pacotes originais intocados, realizando as alterações através da inclusão de novos pacotes. Desta maneira é possível usufruir das atualizações de segurança oferecidos pela distribuição base.

Para realizar a seleção das inúmeras distribuições GNU / Linux existentes foram criados alguns critérios de seleção, que têm por intento não apenas facilitar a tarefa de seleção, mas também torná-la mais objetiva.

Embora o processo de customização seja baseado na alteração de partes que não estejam de acordo com o esperado, a escolha da distribuição base representa um importante passo. Alguns fatores não podem ser simplesmente modificados, e por isso é

essencial que a distribuição escolhida esteja conforme determinadas características básicas, vistas a seguir.

Software Livre

Um dos maiores motivos para o sucesso do GNU / Linux está no fato deste tratar-se de software livre. A expressão "software livre" começou a ser usada pela GNU para representar liberdade no mundo digital, e há uma diferença entre software livre e software grátis. Conforme visto em [3] um software é considerado livre quando atende às quatro premissas básicas seguintes:

- liberdade de execução;
- liberdade de estudar como o programa funciona e poder modificá-lo – esta liberdade implica no acesso ao código fonte;
- liberdade de redistribuir cópias;
- liberdade de aperfeiçoar o software e redistribuir seus aperfeiçoamentos.

As distribuições GNU / Linux que incluem software proprietário foram descartadas do processo de escolha. Da mesma forma foram excluídas as distribuições que não podem ser obtidas de forma totalmente gratuita.

O uso de software livre e de distribuições que não acarretem gastos financeiros justifica-se não apenas por tratar-se de um trabalho acadêmico, mas também para lograr um produto com o menor custo possível.

Linux Standard Base

Nos últimos anos houve uma explosão no número de distribuições GNU / Linux disponíveis. Este crescimento caótico trouxe vários benefícios, sendo o principal a disseminação do GNU / Linux como sistema operacional nos mais variados ambientes. Apesar de promover a popularização, este crescimento também trouxe vários problemas.

Talvez o principal incômodo causado pela inúmera quantidade de diferentes distribuições seja a incompatibilidade entre elas. Em termos práticos isto significa que um determinado programa pode executar normalmente em uma distribuição, mas pode apresentar problemas em outra. Na maioria dos casos o empecilho está na incompatibilidade binária entre os compiladores de diferentes versões, que geram interfaces binárias incompatíveis.

Para minimizar os problemas de incompatibilidade, foi criado o Linux Standard Base Project, ou LSB. O LSB é um conjunto de definições e padrões que criam uma interface única e permitem que uma aplicação rode em qualquer distribuição conformante a estas regras.

Após a criação do LSB as principais distribuições Linux vêm tentando se enquadrar nas normas definidas, no intuito de criar a desejada independência de distribuição. Ser compatível com as normas do LSB não é um pré-requisito para a escolha da distribuição base, mas com certeza é uma característica muito bem-vista.

Gerenciador de Pacotes

Já vimos que o Linux em si, e aqui entende-se como kernel, não possui muita aplicabilidade em um desktop se instalado isoladamente. Por ser um Sistema Operacional ele provê serviços que serão utilizados pelos programas, e não possui muita serventia se estiver desprovido destes.

No princípio esses programas eram distribuídos da mesma forma que o próprio kernel, apenas como código fonte. Era preciso então que o usuário realizasse uma cansativa série de tarefas para vê-lo funcionando. Embora essa rotina funcione, ela tem algumas características altamente indesejáveis, como complexidade na instalação de um simples programa, facilidade de cometer erros ao ter que alterar configurações durante a instalação e quantidade infindável de repetições. E assim o desenvolvedor também sofria com essa metodologia arcaica.

Com a evolução no processo de instalação de contratempos problemas de mais alto nível começaram a surgir. Programas que precisavam de outros programas para rodar também apresentavam dificuldades na instalação, pois checar dependências é uma tarefa cansativa e ineficiente. E podemos citar também os conflitos entre programas que não podem coexistir na mesma máquina.

Para tentar solucionar todos esses problemas foi introduzido o conceito de pacote (package), que pára de tratar um programa como um objeto isolado, adicionando conceitos como dependências e conflitos. Na prática um pacote é um arquivo que contém, além dos códigos binários, metadados que

descrevem o conteúdo dele próprio. Dessa forma um pacote pode prover informações do seu conteúdo e suas relações com outros pacotes.

Para efetuar a tarefa de controle dos pacotes foi introduzido o conceito de Sistema Gerenciador de Pacotes (Package Management System), que se resume a um sistema que mantém informações sobre pacotes instalados e pode instalar novos pacotes, realizar atualizações e remover pacotes já existentes.

É importante salientar que o gerenciamento de pacotes é um dos principais fatores observados na escolha de uma distribuição base.

Instalador

A instalação, uma tarefa imprescindível e de grande importância, é a principal responsável por afugentar usuários inseguros na hora de trocar de Sistema Operacional.

Desde métodos prosaicos que exigem que todos os hardwares sejam selecionados via uma interface de texto até instaladores quase totalmente automáticos com autodetecção de hardware e interface gráfica, a instalação de um sistema GNU / Linux pode variar imensamente de distribuição para distribuição.

Para os fins delimitados por este trabalho, distribuições com o instaladores que seguem a primeira descrição acima apresentada serão descartado.

Documentação e Assistência

Como o GNU / Linux não é um produto de uma empresa e é distribuído livremente, não há um manual oficial que o acompanhe, tão pouco assistência técnica por parte do fabricante. Sendo assim, a documentação disponibilizada é indispensável, pois é a fonte de onde as perguntas serão respondidas.

A documentação é feita do mesmo modo que seu código: através de parceria entre diversos programadores e entidades ligadas ao mundo do software livre. Há o Projeto de Documentação Linux (Linux Documentation Project – LDP) cujo objetivo é servir de base de dados para quem deseja procurar qualquer tipo de informação sobre sistemas GNU / Linux. Essa documentação abrange todo e qualquer tipo de procedimento usual realizado em um sistema.

Mesmo com todas essas formas de documentação, cada distribuição possui peculiaridades e atributos próprios. Logo, informações ou procedimentos genéricos podem não convergir para o mesmo resultado em distribuições distintas. Este é um dos grandes motivos pelo sucesso de uma ou outra distribuição GNU / Linux: a facilidade que o usuário tem em encontrar informações que o ajudem em determinada ocasião.

Essas informações específicas para cada distribuição vão desde tabelas de compatibilidade de hardwares até roteiros com ajuda na hora da instalação e fazem parte do grupo de características que contam na avaliação de uma distribuição.

Distribuição Escolhida

Levando-se em consideração os fatores citados anteriormente e as necessidade impostas, a distribuição escolhida como base para a realização do trabalho foi o Fedora Core.

O Fedora Project, ao lado do Red Hat Enterprise Linux, é um dos descendentes do extinto Red Hat Linux e possui esse nome por tratar-se de um projeto, e não de uma distribuição final. O desenvolvimento do Fedora Project é feito em um fórum público e versões periódicas do Fedora Core (a distribuição em si) são lançadas de tempos em tempos. Embora seja aberto e utilize somente software livre a equipe de engenheiros da Red Hat participa do Fedora Project, porém sem suporte de qualquer tipo proveniente da Red Hat Software; apenas fóruns e grupos de discussões são disponibilizados para esclarecimentos e ajuda aos usuários.

O primeiro e principal motivo pelo qual o Fedora Core foi escolhido como distribuição base é a política adotada pelo Fedora Project, que tem objetivos bem definidos e condizentes com aqueles definidos para este trabalho. O mais importante deles talvez seja o de utilizar-se somente de software livre.

Outro objetivo colaborante com o trabalho descreve a priorização da usabilidade e da filosofia de “apenas funcionar” (descrito em [8] como “just works” philosophy), utilizando-se sempre das configurações e características padrões e livrando o usuário de todos incômodos possíveis.

Além dos citados, outros objetivos pertinentes do Fedora Project são:

- método fácil de fazer atualizações, com mínimas modificações em arquivos de configuração;
- adoção dos softwares mais populares e mais conhecidos, livrando o usuário de programas desconhecidos;
- promoção de perspectivas globais, com suporte a tantas línguas e localidades geográficas quantas forem possíveis.

Todas essas características ajudaram na escolha do Fedora Core como distribuição base, e economizaram muito do trabalho que seria feito. Escolhendo o Fedora Core como base não é necessário remover nenhum pacote essencial à distribuição que seja proprietário. Por tratar-se de uma distribuição destinada ao usuário final, muitas tarefas são automatizadas, e os menus e atalhos já estão organizados de uma forma satisfatória para um usuário leigo.

Embora o Fedora Core não possua o certificado LSB, o Red Hat Linux 9 – última versão lançada sob esse nome – possui certificação LSB 1.3. Naturalmente, o Fedora Core utiliza como Sistema Gerenciador de Pacotes o RPM, originário do Red Hat Linux.

O programa de instalação utilizado pelo Fedora Core é o Anaconda, o mesmo utilizado pelo Red Hat Linux. Fácil de usar e com uma interface gráfica amigável, o Anaconda possui um menu de ajuda que pode estar em várias línguas disponíveis, e exibe explicações detalhadas sobre cada passo que deve ser dado durante a instalação.

Customização da Distribuição

O principal objetivo da customização é tornar a distribuição o mais amigável possível. O ideal é facilitar cada tarefa para o usuário sem que ocorra perda de flexibilidade, mas nem sempre isso é possível. Reduzir o número de aplicações para que a distribuição inteira possa ser armazenada em um único CD-ROM também traz conseqüências que afetam a diretamente a versatilidade da distribuição.

O nome escolhido para a nova distribuição foi “Winona”, que na linguagem Dakota significa “primogênito”.

Embora um dos objetivos seja tornar a distribuição o menos exigente possível em termos de hardware para poder abranger um maior número de máquinas, é necessário no mínimo um computador com um processador Pentium ou superior, 64MB de memória RAM, 2GB de disco rígido e um drive de CD-ROM para o bom desempenho da Winona. Esses requisitos são exigidos principalmente pelo Metacity, o Gerenciador de Janelas do GNOME e pela suíte de aplicativos para escritório OpenOffice.

Anaconda

O Anaconda pode executar em modo gráfico, com a interação do usuário, ou através de scripts em modo não-atendido, que pode dispensar completamente a interação com o usuário. Este último modo, conhecido como “kickstart mode”, será utilizado como artifício para reduzir a intervenção do usuário durante o processo de instalação,

A customização do instalador Anaconda não se limita somente à aplicação do modo não-atendido, também é preciso personalizar todas as telas, modificar a estrutura de arquivos, adicionar pacotes criados, pacotes de atualizações e pacotes extras, remover os pacotes excedentes e gravar o CD personalizado.

O método kickstart existe para suprir uma necessidade que muitos administradores de sistema possuem, a instalação automática. Após realizada uma instalação típica, com a intervenção do usuário, o Anaconda cria automaticamente um arquivo contendo as respostas para todas as perguntas feitas durante a instalação. Este arquivo, que encontra-se no diretório “/root”, é chamado anaconda-ks.cfg e pode então ser utilizado para fazer a instalação automática nos demais equipamentos.

Como já foi destacado anteriormente, certas decisões não serão tomadas pelo usuário, e uma delas é o particionamento do disco rígido. Como esta distribuição não se propõe a viver em harmonia com outros Sistemas Operacionais, todo o conteúdo do disco é apagado e o Anaconda decide automaticamente como particioná-lo. São criadas três partições, uma de boot, uma para dados e outra para swap. O sistema de arquivos utilizado na partição de dados e de boot é o ext3. A partição de boot ocupa entre 75 MB e 100 MB, dependendo do tamanho do disco rígido.

A resolução do vídeo é definida para 800 x 600 e a senha do usuário root é definida como “fedora”, mas como será visto mais adiante, essa distribuição não se propõe a operar de modo multi-usuário. A linguagem do

sistema é definida para português do Brasil. O dispositivo de rede é configurado para receber um endereço IP via DHCP. O fuso-horário é definido arbitrariamente para o de Brasília.

A única decisão que cabe ao usuário durante todo o processo de instalação é definir qual o mapa do teclado a ser utilizado, já que não existe maneira alguma de fazer a detecção automática dos diferentes modelos de teclados.

O arquivo ks.cfg com todas essas definições deve estar na raiz do CD-ROM.

No controle dos pacotes durante a instalação temos arquivo comps.xml, que define quais são os grupos de instalação, assim como os pacotes pertencentes a cada grupo. Numa instalação típica o usuário pode escolher quais grupos ou quais pacotes individuais devem ser instalados.

Na última versão do Red Hat Linux – e posteriormente no Fedora Core – o arquivo comps.xml progrediu de um simples arquivo texto com algumas tags para um arquivo .xml. Essa evolução facilitou muito a definição de grupos e pacotes neste arquivo e sua respectiva leitura pelo instalador Anaconda. Como foram criados novos pacotes para a distribuição Winona, foi preciso alterar o arquivo comps.xml a fim de que eles fossem incluídos na instalação padrão.

Esta parte do trabalho é a que mais despende tempo, já que o Fedora possui aproximadamente 1600 pacotes, armazenados em três CDs, e a proposta desta distribuição é criar um produto compacto.

O método utilizado para formular a lista dos pacotes da distribuição foi listar todos os pacotes da instalação padrão do Fedora, removendo aqueles que fossem classificados como desnecessários. O critério de remoção de pacotes é tratado mais adiante.

A personalização do CD consiste basicamente em obter uma cópia do CD de instalação do Fedora, copiar todo o seu conteúdo para o disco rígido, adicionar os pacotes criados, substituir os pacotes corrigidos pela equipe de manutenção da distribuição base, modificar os arquivos de configuração e então gravar o CD.

O passo mais simples é alterar o nome “Fedora” para “Winona”, no diretório raiz do CD. Também é preciso alterar o arquivo /isolinux/isolinux.cfg para que o processo de instalação comece prontamente, utilizando o método kickstart.

O passo mais trabalhoso e que mais consome tempo na personalização do CD é a eliminação e adição de pacotes. Muitas vezes este processo é descrito como terrível por lidar com as dependências entre os pacotes. Incluir um pacote pode implicar em incluir vários outros, já que é preciso satisfazer suas dependências. Da mesma maneira, excluir um pacote pode implicar em excluir vários outros, já que certos pacotes podem estar presentes apenas para satisfazer as dependências de outros. Para dificultar ainda mais, a substituição de pacotes já presentes por suas versões atualizadas também apresenta os problemas acima citados.

OpenOffice

A instalação do OpenOffice.org do Fedora Core possui graves problemas com relação aos objetivos desse trabalho. As interfaces de usuários dos programas, assim como o corretor ortográfico, foram traduzidos para o português de Portugal. Não foi incluído suporte para hifenação e a ajuda não foi traduzida para nenhuma variante do português, permanecendo totalmente em inglês.

A melhor maneira de solucionar estes inconvenientes seria recompilar o OpenOffice.org, adicionando o suporte a língua nativa brasileira, já que o código fonte inclui suporte para esta língua. Apesar de simples, esta solução provou ser impraticável, já que a compilação do OpenOffice.org mostrou-se impossível no Fedora Core, por problemas com a máquina virtual java.

No sítio openoffice.org.br é possível encontrar uma versão compilada do OpenOffice.org em português do Brasil. Esta versão não possui hifenação e nem sequer ajuda em português. Entretanto, o principal problema é a incompatibilidade entre esta versão e a instalada pelo Fedora Core, que possui vários itens personalizados, como a localização dos arquivos executáveis, de seus atalhos, além da associação dos tipos de arquivos etc.

Também no openoffice.org.br encontram-se um “dicionário” de hifenação e o projeto de tradução da ajuda para o português do Brasil, que é recente mas já encontra-se em um estado bastante avançado.

Adaptar a versão pré-compilada do openoffice.org.br seria uma tarefa muito trabalhosa, portanto decidiu-se manter a versão original do Fedora. A solução encontrada foi

criar um pacote adicional contendo apenas os arquivos responsáveis pela tradução. A criação de um novo pacote implica na alteração do processo de instalação.

Este novo pacote resulta da união de três componentes distintos: os arquivos de tradução da interface de usuário, a ajuda e os dicionários de hifenação e de correção ortográfica. Nenhum dos arquivos deste pacote possui dependência com arquivos externos ao openoffice.org, como, por exemplo, bibliotecas do sistema, o que possibilita que este pacote seja adicionado a qualquer instalação do openoffice.org.

Internet

A experiência de navegar na Internet pode ser bastante frustrante para um usuário do Fedora Core, já que está distribuição não possui suporte para duas tecnologias atualmente bastante difundidas na Internet: Java 2, da Sun Microsystems, e Macromedia Flash.

Para eliminar estas limitações foram criados dois pacotes extras, um plugin para a máquina virtual Java e outro para o flash-player. Estes dois plugins funcionam apenas no Mozilla, que é o navegador web padrão do Fedora Core. O plugin Java possui um pré-requisito, que é a existência de uma máquina virtual Java, portanto foi necessário a criação de outro pacote. O outro navegador do Fedora Core, o Konqueror, não necessita de um plugin para habilitar o suporte a Java; a existência da máquina virtual é suficiente. Por outro lado, a combinação gcc 3.x, konqueror e flash-player mostrou-se não-factível, por problema de interfaces binárias incompatíveis.

Embora essas duas tecnologias tragam facilidades e avanços para a navegação na Internet, elas não estão incluídas na distribuição base por possuírem licenças não-conformantes com a livre redistribuição.

Autologin

Um das decisões mais difíceis no trabalho foi a adoção do autologin. Deixando de lado as questões de segurança, é bastante difícil avaliar até que ponto o autologin facilita a vida do usuário e até que ponto atrapalha.

A dificuldade encontra-se justamente em determinar quantos serão os usuários do computador. Quando utilizado somente por um usuário o autologin certamente é bem-vindo, mas quando este número aumenta os conflitos nas preferências pessoais geralmente não compensam a facilidade trazida.

Decidiu-se que a distribuição opera em modo "single-user", portanto o autologin foi adotado. Para tanto foi preciso criar um usuário diferente adicional, já que não é aconselhável nem permitido realizar autologin do root. Foi necessário fazer uma pequena alteração no arquivo de configuração do gdm, que é o gestor de início do Fedora Core, e também no script de pós-instalação do Anaconda.

Pacotes excluídos

Como explicado anteriormente, muitos pacotes que não são essenciais para o funcionamento do sistema ou para os objetivos traçados foram eliminados. Nos casos em que existem vários pacotes com funcionalidades similares, foi escolhido apenas um para permanecer na distribuição. Esta eliminação de pacotes desnecessários contribui para a diminuição do tamanho da distribuição e para simplificações como, por exemplo, a redução do número de itens no menu.

Problemas Encontrados

Boa parte dos problemas foi encontrada na customização do programa de instalação do Fedora Core, o Anaconda. Para modificar ou até mesmo utilizar de maneira eficiente um software complexo, como é o caso do Anaconda, é fundamental que exista algum tipo de material de apoio, como tutoriais ou artigos baseados em tarefas. Preferencialmente, estes documentos devem ser aprovados pelos desenvolvedores do software. Uma boa indicação de aprovação é a disponibilização dos documentos no site do software.

Apesar de já existir um projeto de documentação do Fedora Project, incluindo todos os seus componentes, a documentação tanto para usuários finais quanto para desenvolvedores praticamente não existe no site do Fedora Project.

Através de ferramentas de procura na Internet, foi possível encontrar alguns artigos escritos por desenvolvedores que também

sentiram falta de documentação oficial. Estes artigos geralmente cobriam as tarefas mais simples, como gravar um CD com o Anaconda. Como principal alternativa para a ausência de documentação oficial, existe uma lista de discussão para desenvolvedores da Fedora e de seus componentes.

É importante salientar que o Anaconda sofreu mudanças estruturais consideráveis recentemente. Estas alterações tornaram inútil grande parte da documentação existente sobre o assunto, tanto na lista de discussão quanto nos artigos publicados.

Na parte de modificação / inclusão de pacotes, as principais dificuldades ficaram por parte das incompatibilidades. Tanto novos aplicativos quanto novas versões de aplicativos existentes podem apresentar incompatibilidades que, ao tentarem ser resolvidas modificando-se o arquivo já existente, e não o novo, provocam incompatibilidades em cadeia, tornando o processo lento e altamente ineficiente.

Conclusões e Trabalhos Futuros

Apesar do esforço despendido, nem todos os objetivos traçados foram alcançados de forma plena. Não foi possível colocar todos os pacotes em apenas um CD. Também não foi possível livrar totalmente a distribuição de software proprietário.

Criar uma distribuição sem tomar outra como base provou ser ineficaz, pois a relação trabalho / benefício não é boa. O ideal é deixar para a distribuição base a inacabável tarefa de manter os pacotes livres de vulnerabilidades.

Como principal trabalho futuro podemos citar a manutenção da distribuição. A manutenção da distribuição engloba atualizações de pacotes, adição de novos pacotes que forem surgindo e até mesmo atualização da distribuição base.

Além disso é preciso criar uma opção para operar no modo sem mídia, para que até mesmo os equipamentos mais modestos possam executar aplicativos que requerem maiores recursos, como o OpenOffice.

[1] AKKERMAN, Wichert. Linux Package Management. 2000. Disponível em: <http://www.wiggy.net/presentations/>. Acesso em: dezembro de 2003.

[2] FREE SOFTWARE FOUNDATION. GNU General Public License. Disponível em: <http://www.gnu.org/copyleft/gpl.html>. Acesso em: novembro de 2003.

[3] FREE SOFTWARE FOUNDATION. The Free Software Definition. Disponível em: <http://www.gnu.org/philosophy/free-sw.html>. Acesso em: dezembro de 2003.

[4] FREE STANDARD GROUP. Linux Standard Base Specification 1.3. Disponível em: <http://www.linuxbase.org/spec/oldspecs.shtml>. Acesso em: dezembro de 2003.

[5] OLIVEIRA, Rômulo Silva de; CARISSIMI, Alexandre da Silva; TOSCANI, Simão Sirineo. Sistemas Operacionais. 2ª edição. Porto Alegre: Sagra Luzzato, 2001.

[6] RED HAT INC. Red Hat Linux Customization Guide. 2003. Disponível em: <http://www.redhat.com/docs/manuals/linux/RHL-9-Manual>. Acesso em: dezembro de 2003.

[7] RED HAT INC. The Anaconda-devel-list Archives. Disponível em: <https://listman.redhat.com/archives/anaconda-devel-list>. Acesso em: dezembro de 2003

[8] RED HAT INC. The Fedora Project Objectives. Disponível em: <http://fedora.redhat.com/about/objectives.html>. Acesso em: dezembro de 2003.

[9] STALLMAN, Richard. Linux and the GNU Project. 2003. Disponível em: <http://www.gnu.org/gnu/linux-and-gnu.html>. Acesso em: dezembro de 2003

8.2 PROCEDIMENTO PADRÃO PARA GRAVAÇÃO DO CD.

```
cp kernel* glibc joe booty
rm vnc-server

export distname="horowitz"
export PRODUCTNAME=Winona
export PYTHONPATH=/usr/lib/anaconda
export PATH="$PATH:/usr/lib/anaconda-runtime"
export RHBASE=/usr/src
mkdir -p $RHBASE/fedora/i386
cd $RHBASE/fedora
for i in 1 2 3 ; do
      mount -o loop yarrow-i386-disc1.iso /mnt ; cd /mnt ; tar -cf - * | ( cd
$RHBASE/fedora/i386/ ; tar -xpf - ) ; cd $RHBASE/fedora; umount /mnt;
done

cd $RHBASE/fedora/i386

mv Fedora $PRODUCTNAME
mkdir -p $RHBASE/fedora/i386/$PRODUCTNAME/SRPMS/

find $RHBASE/fedora/i386 -name TRANS.TBL -exec rm -f {} \;

genhdlist --productpath $PRODUCTNAME $RHBASE/fedora/i386

pkgorder $RHBASE/fedora/i386 i386 $PRODUCTNAME | tee $RHBASE/fedora/pkgorder.txt

buildinstall --comp dist-1 --pkgorder $RHBASE/fedora/pkgorder.txt --version 1 --
product $PRODUCTNAME --release "1.0" --prodpath $PRODUCTNAME $RHBASE/fedora/i386

RELEASE="$PRODUCTNAME ($distname)"

splittree.py --arch=i386 --total-discs=4 --bin-discs=2 --src-discs=2 --release-
string="$RELEASE" --pkgorderfile=$RHBASE/fedora/pkgorder.txt --
distdir=$RHBASE/fedora/i386 --srcdir=$RHBASE/fedora/i386/$PRODUCTNAME/SRPMS/ --
productpath=$PRODUCTNAME

rm -f $RHBASE/fedora/i386-disc1/$PRODUCTNAME/base/hdlist*

genhdlist --withnumbers --fileorder $RHBASE/fedora/pkgorder.txt --productpath
$PRODUCTNAME $RHBASE/fedora/i386-disc1

echo "label instalar" >> $RHBASE/fedora/i386-disc1/isolinux/isolinux.cfg
echo "  kernel vmlinuz" >> $RHBASE/fedora/i386-disc1/isolinux/isolinux.cfg
echo "  append initrd=initrd.img ramdisk_size=8192 ks=cdrom:/ks.cfg" >>
$RHBASE/fedora/i386-disc1/isolinux/isolinux.cfg

myname='Ulisses Muniz de Queiroz <ulissesq@terra.com.br>'
bootimg="isolinux/isolinux.bin"
bootcat="isolinux/boot.cat"
distvers="1"
mkisopts="-R -J -T"
bootopts="-no-emul-boot -boot-load-size 4 -boot-info-table"
mydate="$(date '+%d %b %Y')"
mkisofs $mkisopts $bootopts \
-V "$PRODUCTNAME $distver ($distname) Disk 1" \
-A "$PRODUCTNAME $distver ($distname) created on $mydate" \
-P "$myname" \
-p "$myname" \
-b "$bootimg" \
-c "$bootcat" \
-x lost+found \
-o "$distname-i386-disc1.iso" \
i386-disc1
```

```
implantisomd5 $distname-i386-disc1.iso

myname='Ulisses Muniz de Queiroz <ulissesq@terra.com.br>'
distvers="1"
mkisopts="-R -J -T"
mydate="$(date '+%d %b %Y')"
mkisofs -V "$PRODUCTNAME $distver ($distname) Disk 2" \
-A "$PRODUCTNAME $distver ($distname) created on $mydate" \
-P "$myname" \
-p "$myname" \
-x lost+found \
-o "$distname-i386-disc2.iso" \
i386-disc2

implantisomd5 $distname-i386-disc2.iso
```

8.3 KS.CFG

```
# Kickstart file automatically generated by anaconda.

install
cdrom
lang pt_BR.UTF-8
langsupport --default pt_BR.UTF-8 pt_BR.UTF-8
xconfig --hsync 30.0-70.0 --vsync 50.0-160.0 --resolution 800x600 --depth 16 --
startxonboot  --defaultdesktop gnome
rootpw --iscrypted $1$.BTmS.I2$RnTyzfn4SmsE9ZjumYGhh.
firewall --disabled
authconfig --enableshadow --enablemd5
timezone America/Sao_Paulo
bootloader --location=mbr --append hdc=ide-scsi rhgb

%packages
@ office
@ brazilian-support
@ dialup
@ sound-and-video
@ portuguese-support
@ editors
@ base-x
@ gnome-desktop
@ graphics
@ printing
@ games
@ graphical-internet
kernel
grub

%post
useradd winona
echo winona | passwd --stdin winona
```

8.4 ISOLINUX.CFG

```
default linux
prompt 1
timeout 600
display boot.msg
F1 boot.msg
F2 options.msg
```

```
F3 general.msg
F4 param.msg
F5 rescue.msg
F7 snake.msg
label linux
  kernel vmlinuz
  append initrd=initrd.img ramdisk_size=8192
label text
  kernel vmlinuz
  append initrd=initrd.img text ramdisk_size=8192
label expert
  kernel vmlinuz
  append expert initrd=initrd.img ramdisk_size=8192
label ks
  kernel vmlinuz
  append ks initrd=initrd.img ramdisk_size=8192
label lowres
  kernel vmlinuz
  append initrd=initrd.img lowres ramdisk_size=8192
label instalar
  kernel vmlinuz
  append initrd=initrd.img ramdisk_size=8192 ks=cdrom:/ks.cfg
```

8.5 PROGRAMA PRA LISTAR PACOTES

```
#include <iostream>
#include <fstream>
#include <stdlib.h>

#define BUFFER_SIZE 77
#define STRING_LENGTH 46

using namespace std;

int main (int argc, char *argv[]) {
      char buffer[BUFFER_SIZE];
      int index;
      string pkgname;
      if (argc != 3) {
            cout << "\n\nUsage: " << argv[0] << " {source dir} {dest dir}\n\n\n";
            return 1;
      }
      string destdir(argv[2]);
      string sourcedir(argv[1]);
      ifstream pkgfile("packages.txt");
      ofstream script("pkgcp.sh");
      script << "#!/bin/bash" << endl << endl;
      script << "cd " << sourcedir << endl;
      while (!pkgfile.eof()) {
            pkgfile.getline(buffer, BUFFER_SIZE);
            pkgname = buffer;
            index = pkgname.find_first_of(" ", 1);
            if (index > 1) {
                  pkgname.erase(index, BUFFER_SIZE);
                  script << "cp " << pkgname << "* " << destdir << endl;
            }
      }
      script.close();
      pkgfile.close();
      return 0;
}
```

8.5.1 Makefile

```
CXX=g++
ARCFLAGS=-march=pentium4 -mfpmath=sse,387
OPTFLAGS=-O3 -s
CXXFLAGS=$(ARCFLAGS) $(OPTFLAGS) -pipe
DSTDIR=/tmp/fedora
SRCDIR=/mnt/fedora/usr/src/fedora/i386/Corinna/RPMS
TOPDIR=.

all: copy

clean:
      rm -f pkgcp*

pkgcp: package_copy.cpp
      $(CXX) $(CXXFLAGS) package_copy.cpp -o pkgcp

copy: pkgcp
      $(TOPDIR)/pkgcp $(SRCDIR) $(DSTDIR)
      chmod 755 $(TOPDIR)/pkgcp.sh
      sh $(TOPDIR)/pkgcp.sh
```

8.6 ARQUIVOS .SPEC

8.6.1 openoffice.org-i18n-pt_BR.spec

```
%define oover 1.1.0

Name: openoffice.org-i18n-pt_BR
Version: 1.1rc4
Release: 1
Summary: Brazilian Portuguese support for OpenOffice.org.
Summary(pt_BR): Suporte a português do Brasil para o OpenOffice.org.
Group: Applications/Productivity
Group(pt_BR): Aplicativos/Produtividade
License: LGPL
URL: http://www.openoffice.org
Source0: http://www.openoffice.org.br/openoffice/localized/pt-
br/%{version}/OOo_%{version}_LinuxIntel_install.pt-br.tar.gz
Source1:
http://www.openoffice.org.br/openoffice/contrib/dictionaries/hyph_pt_BR.zip
Source2: http://people.openoffice.org.br/filhocf/teste/Help_%{oover}_pt-BR.zip
Requires: openoffice.org = %{oover}
BuildRoot: %{_tmppath}/%{name}-%{version}-root

%description
Brazilian Portuguese support for OpenOffice.org. This package includes help,
dictionary and hyphenation.

%description -l pt_BR
Suporte a português do Brasil para o OpenOffice.org. Este pacote contém ajuda,
dicionário e hifenação.

%prep
rm -rf pt-BR hyph_pt_BR.dic
%setup -q -n normal

%install
rm -rf %{buildroot}
%{__mkdir_p}
%{buildroot}%{_libdir}/openoffice/{help,program/resource,share/dict/ooo}
./install --prefix=%{buildroot} --single
cd %{buildroot}/OpenOffice.org%{oover}/program/resource/
cp *55.res %{buildroot}%{_libdir}/openoffice/program/resource/
```

```
cd %{buildroot}/OpenOffice.org%{oover}/share/dict/ooo/
cp *pt_BR* %{buildroot}%{_libdir}/openoffice/share/dict/ooo/
cd %{buildroot}
unzip -q %{SOURCE1} hyph_pt_BR.dic -d
%{buildroot}%{_libdir}/openoffice/share/dict/ooo/
unzip -q %{SOURCE2} -d %{buildroot}%{_libdir}/openoffice/help/
rm -rf %{buildroot}/OpenOffice.org%{oover}/

%post
if ! grep pt_BR %{_libdir}/openoffice/share/dict/ooo/dictionary.lst > /dev/null
then
      echo "DICT pt BR pt_BR" >> %{_libdir}/openoffice/share/dict/ooo/dictionary.lst
      echo "HYPH pt BR hyph_pt_BR" >>
%{_libdir}/openoffice/share/dict/ooo/dictionary.lst
fi

%preun
sed -i -e 's/DICT pt BR pt_BR//' -e 's/HYPH pt BR hyph_pt_BR//'
%{_libdir}/openoffice/share/dict/ooo/dictionary.lst

%clean
rm -rf %{buildroot}

%files
%defattr(-,root,root)
%{_libdir}/openoffice/help/*
%{_libdir}/openoffice/program/resource/*
%{_libdir}/openoffice/share/dict/ooo/*
```

8.6.2 desktop-backgrounds.spec

```
%define rh_backgrounds_version 6

Summary: Desktop backgrounds.
Name: desktop-backgrounds
Version: 2.0
Release: 19
License: LGPL
Group: Applications/Multimedia
Source: redhat-backgrounds-%{rh_backgrounds_version}.tar.bz2
Source2: Propaganda-1.0.0.tar.gz
Source3: README.Propaganda
## Source4: beta-placeholder.png
BuildRoot: %{_tmppath}/%{name}-%{version}-root
BuildArchitectures: noarch

%description
The desktop-backgrounds package contains artwork intended
to be used as desktop wallpaper.

%package basic

Summary: Desktop background base set.
Group: Applications/Multimedia

Provides: desktop-backgrounds
Obsoletes: desktop-backgrounds

%description basic
The desktop-backgrounds-basic package contains a good basic set of
images to use for your desktop background.

%package extra

Summary: Desktop background images.
Group: Applications/Multimedia
```

```
%description extra
The desktop-backgrounds-extra package contains a larger set of images
to use for your desktop background. It builds on
desktop-backgrounds-basic.

%prep
%setup -n redhat-backgrounds-%{rh_backgrounds_version}

# move things where %doc can find them
cp %{SOURCE3} .
mv images/space/*.ps .
mv images/space/README* .

# add propaganda
(cd tiles && tar zxf %{SOURCE2})

## put in placeholder for the beta
## cp -f %{SOURCE4} images/default.png

%install
rm -rf $RPM_BUILD_ROOT

mkdir -p $RPM_BUILD_ROOT%{_prefix}/share/backgrounds
cd $RPM_BUILD_ROOT%{_prefix}/share/backgrounds

cp -a $RPM_BUILD_DIR/redhat-backgrounds-%{rh_backgrounds_version}/images .
cp -a $RPM_BUILD_DIR/redhat-backgrounds-%{rh_backgrounds_version}/tiles .

%clean
rm -rf $RPM_BUILD_ROOT

# basic contains some reasonable sane basic tiles
%files basic
%defattr(-, root, root)
%dir %{_datadir}/backgrounds
%dir %{_datadir}/backgrounds/tiles
%dir %{_datadir}/backgrounds/images
%{_datadir}/backgrounds/tiles/*.png
%{_datadir}/backgrounds/tiles/*jpg
%{_datadir}/backgrounds/images/default.png

# extra contains big images, plus Propaganda tiles
%files extra
%defattr(-, root, root)
%doc README.space PHOTO_FAQ.ps README.Propaganda
%dir %{_datadir}/backgrounds
%dir %{_datadir}/backgrounds/images
%dir %{_datadir}/backgrounds/tiles
%{_datadir}/backgrounds/tiles/Propaganda
%{_datadir}/backgrounds/images/*
## we'll see if rpm likes this
%exclude %{_datadir}/backgrounds/images/default.png

%changelog
* Fri Jan 30 2004 Ulisses Muniz de Queiroz <ulissesq@terra.com.br>
- replaced default background with Crineu Tres' artwork

* Sun Nov  2 2003 Elliot Lee <sopwith@redhat.com> 2.0-18
- redhat-backgrounds-6

* Wed Oct 29 2003 Havoc Pennington <hp@redhat.com> 2.0-17
- redhat-backgrounds-5
```

8.6.3 java2.spec

```
%define major    1
%define minor    4
```

```
%define sub 2
%define release  03

Name: java2
Version: %{major}.%{minor}.%{sub}
Release: %{release}
Summary: Java(TM) 2 Software Virtual Machine.
Summary(pt_BR): MÃ¡quina Virtual Java(TM) 2.
Group: System Environment/Base
Group(pt_BR): Ambiente do Sistema/Base
License: 1994-2001 Sun Microsystems, Inc.
URL: http://java.sun.com
Source: j2sdk-%{major}_%{minor}_%{sub}_%{release}-linux-i586.bin
BuildRequires: binutils coreutils gcc glibc make
BuildRoot: %{_tmppath}/%{name}-%{version}-root

%description
Desktop personal computers are more interactive and exciting thanks to Java
technology. By enabling all types of programs to run on just about any machine,
regardless of operating system, Java technology brings you the best the digital
world has to offer, from multiplayer role-playing fantasy games to detailed
market tracking applications.

%description -l pt_BR
Computadores pessoais estÃ£o mais interativos e divertidos graÃ§as a tecnologia
Java. Ao permitir que todos os tipos de programas rodem em qualquer tipo de mÃ¡-
quina, independentemente do sistema operacional, a tecnologia Java trÃ¡s atÃ© vocÃª
o melhor que o mundo digital tem para oferecer, desde jogos role-playing multi-
player atÃ© aplicaÃ§Ãµes de rastreamento detalhado de marketing.

%package plugin
Summary: Enables applets to be run within a web browser.
Summary(pt_BR): Permite que applets rodem dentro de um navegador web.
Group: Applications/Internet
Group(pt_BR): AplicaÃ§Ãµes/Internet
Requires: java2 = %{version}

%description plugin
Java Plug-in technology establishes a connection between popular browsers and
the Java platform. This connection enables applets on Web sites to be run within
a browser on the desktop.

%description plugin -l pt_BR
A tecnologia Java Plug-in estabelece uma conexÃ£o entre navegadores de Internet
populares e a plataforma Java. Esta conexÃ£o permite que applets em pÃ¡ginas da
Internet executem dentro de um navegador.

%package sdk
Summary: Java(TM) 2 Software Development Kit, Standard Edition.
Summary(pt_BR): Kit de Desenvolvimento de Software Java(TM) 2, EdiÃ§Ã£o PadrÃ£o.
Group: Development/Tools
Group(pt_BR): Desenvolvimento/Ferramentas
Requires: java2 = %{version}

%description sdk
The Java 2 SDK, Standard Edition includes tools used by programmers to develop
Java software applets and applications. The SDK also provides the foundation for
IDE (Integrated Development Environment) tools such as Sun's Forte for Java,
Community Edition, the Java(TM) 2 Platform, Enterprise Edition (J2EE), Java-
based application servers and more. The Java 2 Software Development Kit, SDK, is
a development environment for building applications, applets, and components
that can be deployed on the Java platform. The Java 2 SDK software includes
tools useful for developing and testing programs written in the Java programming
language and running on the Java platform. These tools are designed to be used
from the command line. Except for appletviewer, these tools do not provide a
graphical user interface.

%description sdk -l pt_BR
```

```
O Java 2 SDK, Standard Edition inclui ferramentas usadas para programadores
desenvolverem applets Java e aplicaÃ§Ãµes. O SDK tambÃ©m provÃª a base para ferra-
mentas IDE (Ambiente de desenvolvimento Integrado) como o Forte for Java, Com-
munity Edition, da Sun, o Java(TM) 2 Platform, Enterprise Edition (J2EE), ser-
vidores de aplicaÃ§Ãµes em Java e mais. O Java 2 Software Development Kit, SDK,
Ã© um ambiente de desenvolvimento para construÃ§Ã£o de aplicativos, applets e com-
ponentes que podem ser implantados na plataforma Java. O Java 2 SDK inclui fer-
ramentas Ãºteis para desenvolvimento e teste de programas escritos na linguagem
de programaÃ§Ã£o Java e que executem na plataforma Java. Estas ferramentas sÃ£o
projetadas para serem utilizadas na linha de comando. Exceto o appletviewer,
estas ferramentas nÃ£o possuem uma interface de usuÃ¡rio grÃ¡fica.

%prep
%{SOURCE0} << EOF
yes
EOF

%install
rm -rf %{buildroot}
%{__mkdir_p} %{buildroot}%{_prefix}/java
%{__mkdir_p} %{buildroot}%{_libdir}/mozilla/plugins/
cd %{buildroot}%{_prefix}/java/
mv %{_builddir}/j2sdk%{version}_%{release} .
mv j2sdk%{version}_%{release}/jre/ j2re%{version}_%{release}/
%{__ln_s} j2sdk%{version}_%{release} j2sdk
%{__ln_s} j2re%{version}_%{release} j2re
%{__ln_s}f ../j2re%{version}_%{release} j2sdk%{version}_%{release}/jre
%{__ln_s}f %{_prefix}/java/j2re/plugin/i386/ns610-gcc32/libjavaplugin_oji.so
%{buildroot}%{_libdir}/mozilla/plugins/

%clean
rm -rf %{buildroot}

%files
%defattr(-,root,root)
%dir %{_prefix}/java/
%{_prefix}/java/j2re*/

%files plugin
%defattr(-,root,root)
%{_libdir}/mozilla/plugins/*

%files sdk
%defattr(-,root,root)
%{_prefix}/java/j2sdk*/
```

8.6.4 gdm.spec

```
%define pango_version 1.2.0
%define gtk2_version 2.2.0
%define libglade2_version 2.0.0
%define libgnomeui_version 2.2.0
%define libgnomecanvas_version 2.0.0
%define librsvg2_version 2.0.1
%define libxml2_version 2.4.21
%define scrollkeeper_version 0.3.4
%define pam_version 0.75
%define desktop_file_utils_version 0.2.90
%define gail_version 1.2.0
%if %{?WITH_SELINUX:0}%{!?WITH_SELINUX:1}
%define WITH_SELINUX 0
%endif

Summary: The GNOME Display Manager.
Name: gdm
Version: 2.4.4.5
Release: 2
```

```
Epoch: 1
License: LGPL/GPL
Group: User Interface/X
Source: ftp://ftp.gnome.org/pub/GNOME/sources/gdm-%{PACKAGE_VERSION}.tar.bz2
URL: ftp://ftp.gnome.org/pub/GNOME/sources/gdm/
Source1: gdm.conf
## FIXME: is this relevant?
## temporary ja.po hack for date format
Source7: gdm-ja.po

Patch1: gdm-2.4.4.3-rhconfig.patch
## we're going to try UTF-8 CJK
## Patch2: gdm-2.4.1.1-cjk-no-utf8.patch
Patch4: gdm-2.4.2.102-pam_timestamp.patch
## there's no greek font so don't translate greek in language picker,
## it looks awful
Patch11: gdm-2.4.0.7-nogreek.patch

BuildRoot: %{_tmppath}/gdm-%{PACKAGE_VERSION}-root

Prereq: /usr/sbin/useradd
Prereq: /usr/bin/scrollkeeper-update
Requires: gtk2 >= %{gtk2_version}
Requires: libglade2 >= %{libglade2_version}
Requires: libgnomeui >= %{libgnomeui_version}
Requires: libgnomecanvas >= %{libgnomecanvas_version}
Requires: librsvg2 >= %{librsvg2_version}
Requires: libxml2 >= %{libxml2_version}
Requires: pam >= %{pam_version}
Requires: /etc/pam.d/system-auth
Requires: /etc/X11/xdm/Xsession
Requires: usermode
Requires: xinitrc >= 3.33-1
Requires: xsri >= 1:2.0.2
Requires: /sbin/nologin
Requires: redhat-artwork >= 0.9
Requires: /usr/share/desktop-menu-patches/gnome-gdmsetup.desktop
BuildRequires: scrollkeeper >= %{scrollkeeper_version}
BuildRequires: pango-devel >= %{pango_version}
BuildRequires: gtk2-devel >= %{gtk2_version}
BuildRequires: libglade2-devel >= %{libglade2_version}
BuildRequires: libgnomeui-devel >= %{libgnomeui_version}
BuildRequires: libgnomecanvas-devel >= %{libgnomecanvas_version}
BuildRequires: librsvg2-devel >= %{librsvg2_version}
BuildRequires: libxml2-devel >= %{libxml2_version}
BuildRequires: usermode
BuildRequires: pam-devel >= %{pam_version}
BuildRequires: fontconfig
BuildRequires: desktop-file-utils >= %{desktop_file_utils_version}
BuildRequires: gail-devel >= %{gail_version}
BuildRequires: libgsf-devel
BuildRequires: libtool automake autoconf
BuildRequires: libcroco-devel

%description
Gdm (the GNOME Display Manager) is a highly configurable
reimplementation of xdm, the X Display Manager. Gdm allows you to log
into your system with the X Window System running and supports running
several different X sessions on your local machine at the same time.

%prep
%setup -q

%patch1 -p1 -b .rhconfig
## %patch2 -p1 -b .cjk-no-utf8
%patch4 -p1 -b .pam_timestamp
%patch11 -p1 -b .nogreek
```

```
## put in ja translation
cp -f %{SOURCE7} po/ja.po

%build
aclocal
libtoolize --force --copy
automake --add-missing
autoconf
autoheader
%configure --prefix=%{_prefix} --sysconfdir=/etc/X11 --with-pam-prefix=/etc --
localstatedir=/var --enable-console-helper \
%if %{WITH_SELINUX}
--with-selinux
%else
--without-selinux
%endif
make

%install
[ -n "$RPM_BUILD_ROOT" -a "$RPM_BUILD_ROOT" != / ] && rm -rf $RPM_BUILD_ROOT


make sysconfdir=$RPM_BUILD_ROOT/etc/X11 libdir=$RPM_BUILD_ROOT%{_libdir}\
    libexecdir=$RPM_BUILD_ROOT%{_libexecdir} \
    mandir=$RPM_BUILD_ROOT%{_mandir} \
    prefix=$RPM_BUILD_ROOT%{_prefix} bindir=$RPM_BUILD_ROOT%{_bindir} \
    datadir=$RPM_BUILD_ROOT%{_datadir} \
    localstatedir=$RPM_BUILD_ROOT%{_localstatedir} \
    sbindir=$RPM_BUILD_ROOT%{_sbindir} \
    PAM_PREFIX=$RPM_BUILD_ROOT/etc install

# docs go elsewhere
rm -rf $RPM_BUILD_ROOT/%{prefix}/doc

# change default Init script for :0 to be Red Hat default
ln -sf ../../xdm/Xsetup_0 $RPM_BUILD_ROOT/etc/X11/gdm/Init/:0

# create log dir
mkdir -p $RPM_BUILD_ROOT/var/log/gdm

# remove the gdm Xsession as we're using the xdm one
rm -f $RPM_BUILD_ROOT%{_sysconfdir}/X11/gdm/Xsession

rm -f $RPM_BUILD_ROOT%{_libdir}/gtk-2.0/modules/*.a
rm -f $RPM_BUILD_ROOT%{_libdir}/gtk-2.0/modules/*.la

# remove the gnome session file, since we put that in gnome-session
rm -f $RPM_BUILD_ROOT%{_sysconfdir}/X11/dm/Sessions/gnome.desktop

# no dumb flexiserver thing, Xnest is too broken
# FIXME: flexiserver is NOT just Xnest
#        dual login with standard (non-Xnest) flexiserver is a very
#        often requested feature even though it exists but
#        redhat removes it --George
rm -f $RPM_BUILD_ROOT%{_datadir}/applications/gdmflexi*.desktop

# use patched gdmsetup desktop file
rm -f $RPM_BUILD_ROOT%{_datadir}/applications/gdmsetup.desktop
ln -sf %{_datadir}/desktop-menu-patches/gnome-gdmsetup.desktop
$RPM_BUILD_ROOT%{_datadir}/applications/gnome-gdmsetup.desktop

# fix the "login photo" file
desktop-file-install --vendor gnome --delete-original       \
  --dir $RPM_BUILD_ROOT%{_datadir}/applications             \
  --add-category X-Red-Hat-Base                             \
  $RPM_BUILD_ROOT%{_datadir}/gnome/capplets/*

# broken install-data-local in gui/Makefile.am makes this necessary
```

```
(cd $RPM_BUILD_ROOT%{_bindir} && ln -sf gdmXnestchooser gdmXnest)

rm -rf $RPM_BUILD_ROOT%{_localstatedir}/scrollkeeper

cp -f %{SOURCE1} %{buildroot}%{_sysconfdir}/X11/gdm/

%find_lang gdm-2.4

%clean
[ -n "$RPM_BUILD_ROOT" -a "$RPM_BUILD_ROOT" != / ] && rm -rf $RPM_BUILD_ROOT

%pre
/usr/sbin/useradd -M -u 42 -d /var/gdm -s /sbin/nologin -r gdm > /dev/null 2>&1
/usr/sbin/usermod -d /var/gdm -s /sbin/nologin gdm >/dev/null 2>&1
# ignore errors, as we can't disambiguate between gdm already existed
# and couldn't create account with the current adduser.
exit 0

%post
/sbin/ldconfig
scrollkeeper-update

# Attempt to restart GDM softly by use of the fifo.  Wont work on older
# then 2.2.3.1 versions but should work nicely on later upgrades.
# FIXME: this is just way too complex
FIFOFILE=`grep '^ServAuthDir=' %{_sysconfdir}/X11/gdm/gdm.conf | sed -e
's/^ServAuthDir=//'`
if test x$FIFOFILE = x ; then
      FIFOFILE=%{_localstatedir}/gdm/.gdmfifo
else
      FIFOFILE="$FIFOFILE"/.gdmfifo
fi
PIDFILE=`grep '^PidFile=' %{_sysconfdir}/X11/gdm/gdm.conf | sed -e 's/^PidFile=//'`
if test x$PIDFILE = x ; then
      PIDFILE=/var/run/gdm.pid
fi
if test -w $FIFOFILE && test -f $PIDFILE && kill -0 `cat $PIDFILE` 2>/dev/null ;
then
      (echo;echo SOFT_RESTART) >> $FIFOFILE
fi
# ignore error in the above
exit 0

%postun
/sbin/ldconfig
scrollkeeper-update

%files -f gdm-2.4.lang
%defattr(-, root, root)

%doc AUTHORS COPYING ChangeLog NEWS README TODO

%dir /etc/X11/gdm
# Not sure which package /etc/X11/dm dir should belong to,
# this dir was agreed on among KDM and GDM maintainer to host
# the new session setup
%config /etc/X11/dm/Sessions/*
%config /etc/X11/gdm/gdm.conf
/etc/X11/gdm/factory-gdm.conf
%config /etc/X11/gdm/XKeepsCrashing
%config /etc/X11/gdm/locale.alias
%config /etc/X11/gdm/Init/*
%config /etc/X11/gdm/PostLogin/*
%config /etc/X11/gdm/PreSession/*
%config /etc/X11/gdm/PostSession/*
%config /etc/X11/gdm/modules/*
%config /etc/pam.d/gdm
%config /etc/pam.d/gdmsetup
```

```
%config /etc/pam.d/gdm-autologin
%config /etc/security/console.apps/gdmsetup
%dir /etc/X11/gdm/Init
%dir /etc/X11/gdm/PreSession
%dir /etc/X11/gdm/PostSession
%dir /etc/X11/gdm/PostLogin
%dir /etc/X11/gdm/modules
%{_datadir}/pixmaps
%{_datadir}/gdm
%{_datadir}/xsessions/*
%{_datadir}/applications
%{_datadir}/gnome/help/gdm
%{_datadir}/omf/gdm
%{_libdir}/gtk-2.0/modules/*.so
%{_bindir}/*
%{_libexecdir}/*
%{_mandir}/man*/*
%{_sbindir}/*
%dir %{_localstatedir}/log/gdm

%attr(1770, root, gdm) %dir %{_localstatedir}/gdm

%changelog
* Mon Feb 02 2004 Ulisses Muniz de Queiroz <ulissesq@terra.com.br> 1:2.4.4.5-2
- autologin user winona by default
- add libcroco-devel as build req
```

8.6.5 gaim.spec

```
Name: gaim
Version: 0.74
Release: 6
Epoch: 1
License: GPL
Group: Applications/Internet
URL: http://gaim.sourceforge.net/
Source: http://prdownloads.sourceforge.net/gaim/gaim-%{version}.tar.bz2
Patch0: gaim-desktop.patch
Patch2: gaim-0.68-pofiles.patch
Patch3: gaim-0.71-newtrayicon.patch
Patch4: gaim-0.71-xinput.patch
BuildRoot: %{_tmppath}/%{name}-%{version}-root
Summary: A GTK+ clone of the AOL Instant Messenger client.
Prereq: /sbin/ldconfig
Requires: htmlview, mozilla-nss
BuildRequires: libao-devel, startup-notification-devel, audiofile-devel
BuildRequires: libtool, gtkspell-devel, aspell-devel, mozilla-nss-devel, gtk2-devel
ExclusiveArch: i386 x86_64 ia64 ppc s390

%description
Gaim is a clone of America Online's Instant Messenger client. It
features nearly all of the functionality of the official AIM client
while also being smaller, faster, and commercial-free.

%prep
%setup -q
%patch0 -p1
%patch2 -p1
%patch3 -p1
%patch4 -p1
echo "Name[pt_BR]=Cliente de Mensagens" >> gaim.desktop
echo "Comment[pt_BR]=Cliente de Mensagens InstantÃ¢neas Multi-Protocolo" >>
gaim.desktop

%build
%configure --disable-gnome --disable-artsc
make
```

```
%install
rm -rf $RPM_BUILD_ROOT
make DESTDIR=$RPM_BUILD_ROOT install
mkdir -p $RPM_BUILD_ROOT/%{_datadir}/applications/
# remove libtool libraries and static libraries
rm -f `find $RPM_BUILD_ROOT -name "*.la" -o -name "*.a"`
# remove the old perllocal.pod file
rm $RPM_BUILD_ROOT/%{perl_archlib}/perllocal.pod
# make sure that we can write to all the files we've installed
# so that they are properly stripped
chmod -R u+w $RPM_BUILD_ROOT/*

%find_lang %{name}

%post
/sbin/ldconfig -n %{_libdir}/gaim

%postun
/sbin/ldconfig -n %{_libdir}/gaim

%clean
rm -rf $RPM_BUILD_ROOT

%files -f %{name}.lang
%defattr(-,root,root)
%doc doc/the_penguin.txt doc/CREDITS NEWS COPYING AUTHORS doc/FAQ README ChangeLog
doc/PERL-HOWTO.dox doc/gaims_funniest_home_convos.txt HACKING
#%%config %{_sysconfdir}/CORBA/servers/gaim_applet.gnorba
%{_bindir}/*
%{_libdir}/gaim
%{_includedir}/gaim-remote
%{_libdir}/libgaim*.so
%{_libdir}/libgaim*.so.0.*
%{_mandir}/*/*
#%%{_datadir}/applets/Network/gaim_applet.desktop
%{_datadir}/applications/gaim.desktop
%{_datadir}/pixmaps/gaim
%{_datadir}/pixmaps/gaim.png
%{_datadir}/sounds/gaim
%{_libdir}/perl5/vendor_perl/*/*/*
#%%{_datadir}/pixmaps/gaim/*

%changelog
* Thu Jan 29 2004 Ulisses Muniz de Queiroz <ulissesq@terra.com.br> 1:0.74-6
- Include pt_BR strings to the desktop item.
```

8.6.6 rpm.spec

```
%define     with_python_subpackage 1%{nil}
%define     with_python_version    2.2%{nil}
%define     with_bzip2       1%{nil}
%define     with_apidocs           1%{nil}

# XXX legacy requires './' payload prefix to be omitted from rpm packages.
%define     _noPayloadPrefix 1

%define     __prefix    /usr
%{?!_lib: %define _lib lib}
%{expand: %%define __share %(if [ -d %{__prefix}/share/man ]; then echo /share ;
else echo %%{nil} ; fi)}

%define __bindir %{__prefix}/bin
%define __includedir   %{__prefix}/include
%define __libdir %{__prefix}/%{_lib}
%define __mandir %{__prefix}%{__share}/man

Summary: The RPM package management system.
```

```
Name: rpm
%define version 4.2.1
Version: %{version}
%{expand: %%define rpm_version %{version}}
Release: 0.30
Group: System Environment/Base
Source: ftp://ftp.rpm.org/pub/rpm/dist/rpm-4.0.x/rpm-%{rpm_version}.tar.gz
Copyright: GPL
Conflicts: patch < 2.5
%ifos linux
Prereq: fileutils shadow-utils
%endif
Requires: popt = 1.8.1
Obsoletes: rpm-perl < %{version}

# XXX necessary only to drag in /usr/lib/libelf.a, otherwise internal elfutils.
BuildRequires: elfutils-libelf

BuildRequires: zlib-devel

BuildRequires: beecrypt-devel >= 0:3.0.0-2

BuildRequires: elfutils-devel

Requires: beecrypt >= 0:3.0.0-2

# XXX Red Hat 5.2 has not bzip2 or python
%if %{with_bzip2}
BuildRequires: bzip2 >= 0.9.0c-2
%endif
%if %{with_python_subpackage}
BuildRequires: python-devel >= %{with_python_version}
%endif

BuildRoot: %{_tmppath}/%{name}-root

%description
The RPM Package Manager (RPM) is a powerful command line driven
package management system capable of installing, uninstalling,
verifying, querying, and updating software packages. Each software
package consists of an archive of files along with information about
the package like its version, a description, etc.

%package devel
Summary:  Development files for manipulating RPM packages.
Group: Development/Libraries
Requires: rpm = %{rpm_version}

%description devel
This package contains the RPM C library and header files. These
development files will simplify the process of writing programs that
manipulate RPM packages and databases. These files are intended to
simplify the process of creating graphical package managers or any
other tools that need an intimate knowledge of RPM packages in order
to function.

This package should be installed if you want to develop programs that
will manipulate RPM packages and databases.

%package build
Summary: Scripts and executable programs used to build packages.
Group: Development/Tools
Requires: rpm = %{rpm_version}, patch >= 2.5, file
Provides: rpmbuild(VendorConfig) = 4.1-1

%description build
The rpm-build package contains the scripts and executable programs
that are used to build packages using the RPM Package Manager.
```

```
%if %{with_python_subpackage}
%package python
Summary: Python bindings for apps which will manipulate RPM packages.
Group: Development/Libraries
Requires: rpm = %{rpm_version}
Requires: python >= %{with_python_version}
Requires: elfutils >= 0.55

%description python
The rpm-python package contains a module that permits applications
written in the Python programming language to use the interface
supplied by RPM Package Manager libraries.

This package should be installed if you want to develop Python
programs that will manipulate RPM packages and databases.
%endif

%package -n popt
Summary: A C library for parsing command line parameters.
Group: Development/Libraries
Version: 1.8.1

%description -n popt
Popt is a C library for parsing command line parameters. Popt was
heavily influenced by the getopt() and getopt_long() functions, but it
improves on them by allowing more powerful argument expansion. Popt
can parse arbitrary argv[] style arrays and automatically set
variables based on command line arguments. Popt allows command line
arguments to be aliased via configuration files and includes utility
functions for parsing arbitrary strings into argv[] arrays using
shell-like rules.

%prep
%setup -q
sed -i -e 's;#%distributio;%distributios;g' \
       -e 's;distributiosn;distribution Winona;g' \
       -e 's;{_usrsrc}/redhat;{_usrsrc}/winona;g' macros.in

%build

# XXX rpm needs functioning nptl for configure tests
unset LD_ASSUME_KERNEL || :

%if %{with_python_subpackage}
WITH_PYTHON="--with-python=%{with_python_version}"
%else
WITH_PYTHON="--without-python"
%endif

%ifos linux
%ifarch x86_64 s390 s390x
CFLAGS="$RPM_OPT_FLAGS -fPIC"; export CFLAGS
%else
CFLAGS="$RPM_OPT_FLAGS"; export CFLAGS
%endif
CFLAGS="$RPM_OPT_FLAGS" ./configure --prefix=%{__prefix} --sysconfdir=/etc \
      --localstatedir=/var --infodir='${prefix}%{__share}/info' \
      --mandir='${prefix}%{__share}/man' \
      $WITH_PYTHON --without-javaglue
%else
CFLAGS="$RPM_OPT_FLAGS" ./configure --prefix=%{__prefix} $WITH_PYTHON \
      --without-javaglue
%endif

# XXX hack out O_DIRECT support in db4 for now.
perl -pi -e 's/#define HAVE_O_DIRECT 1/#undef HAVE_O_DIRECT/' db3/db_config.h
```

```
make RPMCANONVENDOR=winona

%install
# XXX rpm needs functioning nptl for configure tests
unset LD_ASSUME_KERNEL || :

rm -rf $RPM_BUILD_ROOT

make DESTDIR="$RPM_BUILD_ROOT" RPMCANONVENDOR=winona install

%ifos linux

# Save list of packages through cron
mkdir -p ${RPM_BUILD_ROOT}/etc/cron.daily
install -m 755 scripts/rpm.daily ${RPM_BUILD_ROOT}/etc/cron.daily/rpm

mkdir -p ${RPM_BUILD_ROOT}/etc/logrotate.d
install -m 644 scripts/rpm.log ${RPM_BUILD_ROOT}/etc/logrotate.d/rpm

mkdir -p $RPM_BUILD_ROOT/etc/rpm

mkdir -p $RPM_BUILD_ROOT/var/spool/repackage
mkdir -p $RPM_BUILD_ROOT/var/lib/rpm
for dbi in \
      Basenames Conflictname Dirnames Group Installtid Name Packages \
      Providename Provideversion Requirename Requireversion Triggername \
      Filemd5s Pubkeys Sha1header Sigmd5 \
      __db.001 __db.002 __db.003 __db.004 __db.005 __db.006 __db.007 \
      __db.008 __db.009
do
    touch $RPM_BUILD_ROOT/var/lib/rpm/$dbi
done

%endif

%if %{with_apidocs}
gzip -9n apidocs/man/man*/* || :
%endif

# Get rid of unpackaged files
{ cd $RPM_BUILD_ROOT
  rm -rf .%{__includedir}/beecrypt
  rm -f .%{__libdir}/libbeecrypt.{a,la,so.2.2.0}
  rm -f
.%{__prefix}/lib/rpm/{Specfile.pm,cpanflute,cpanflute2,rpmdiff,rpmdiff.cgi,sql.prov
,sql.req,tcl.req}
  rm -rf .%{__mandir}/{fr,ko}
}

%clean
rm -rf $RPM_BUILD_ROOT

%pre
%ifos linux
if [ -f /var/lib/rpm/packages.rpm ]; then
    echo "
You have (unsupported)
      /var/lib/rpm/packages.rpm    db1 format installed package headers
Please install rpm-4.0.4 first, and do
      rpm --rebuilddb
to convert your database from db1 to db3 format.
"
    exit 1
fi
/usr/sbin/groupadd -g 37 rpm                  > /dev/null 2>&1
/usr/sbin/useradd  -r -d /var/lib/rpm -u 37 -g 37 rpm -s /sbin/nologin     >
/dev/null 2>&1
%endif
```

```
exit 0

%post
%ifos linux
/sbin/ldconfig
/bin/chown rpm.rpm /var/lib/rpm/[A-Z]*
%endif
exit 0

%ifos linux
%postun
/sbin/ldconfig
if [ $1 = 0 ]; then
    /usr/sbin/userdel rpm
    /usr/sbin/groupdel rpm
fi
exit 0

%post devel -p /sbin/ldconfig
%postun devel -p /sbin/ldconfig

%post -n popt -p /sbin/ldconfig
%postun -n popt -p /sbin/ldconfig
%endif

%if %{with_python_subpackage}
%post python -p /sbin/ldconfig
%postun python -p /sbin/ldconfig
%endif

%define     rpmattr          %attr(0755, rpm, rpm)

%files
%defattr(-,root,root)
%doc RPM-PGP-KEY RPM-GPG-KEY BETA-GPG-KEY CHANGES GROUPS doc/manual/[a-z]*
# XXX comment these lines out if building with rpm that knows not %pubkey attr
%pubkey RPM-PGP-KEY
%pubkey RPM-GPG-KEY
%pubkey BETA-GPG-KEY
%attr(0755, rpm, rpm)  /bin/rpm

%ifos linux
%config(noreplace,missingok) /etc/cron.daily/rpm
%config(noreplace,missingok) /etc/logrotate.d/rpm
%dir                    /etc/rpm
#%config(noreplace,missingok)      /etc/rpm/macros.*
%attr(0755, rpm, rpm)  %dir /var/lib/rpm
%attr(0755, rpm, rpm)  %dir /var/spool/repackage

%define     rpmdbattr %attr(0644, rpm, rpm) %verify(not md5 size mtime) %ghost
%config(missingok,noreplace)
%rpmdbattr  /var/lib/rpm/*
%endif

%rpmattr    %{__bindir}/rpm2cpio
%rpmattr    %{__bindir}/gendiff
%rpmattr    %{__bindir}/rpmdb
#%rpmattr   %{__bindir}/rpm[eiu]
%rpmattr    %{__bindir}/rpmsign
%rpmattr    %{__bindir}/rpmquery
%rpmattr    %{__bindir}/rpmverify

%{__libdir}/librpm-4.2.so
%{__libdir}/librpmdb-4.2.so
%{__libdir}/librpmio-4.2.so
%{__libdir}/librpmbuild-4.2.so

%attr(0755, rpm, rpm)  %dir %{__prefix}/lib/rpm
```

```
%rpmattr    %{__prefix}/lib/rpm/config.guess
%rpmattr    %{__prefix}/lib/rpm/config.sub
%rpmattr    %{__prefix}/lib/rpm/convertrpmrc.sh
%attr(0644, rpm, rpm)  %{__prefix}/lib/rpm/macros
%rpmattr    %{__prefix}/lib/rpm/mkinstalldirs
%rpmattr    %{__prefix}/lib/rpm/rpm.*
%rpmattr    %{__prefix}/lib/rpm/rpm2cpio.sh
%rpmattr    %{__prefix}/lib/rpm/rpm[deiukqv]
%rpmattr    %{__prefix}/lib/rpm/tgpg
%attr(0644, rpm, rpm)  %{__prefix}/lib/rpm/rpmpopt*
%attr(0644, rpm, rpm)  %{__prefix}/lib/rpm/rpmrc

%ifarch i386 i486 i586 i686 athlon
%attr(-, rpm, rpm)          %{__prefix}/lib/rpm/i[3456]86*
%attr(-, rpm, rpm)          %{__prefix}/lib/rpm/athlon*
%endif
%ifarch alpha alphaev5 alphaev56 alphapca56 alphaev6 alphaev67
%attr(-, rpm, rpm)          %{__prefix}/lib/rpm/alpha*
%endif
%ifarch sparc sparcv9 sparc64
%attr(-, rpm, rpm)          %{__prefix}/lib/rpm/sparc*
%endif
%ifarch ia64
%attr(-, rpm, rpm)          %{__prefix}/lib/rpm/ia64*
%endif
%ifarch powerpc ppc ppciseries ppcpseries ppcmac ppc64
%attr(-, rpm, rpm)          %{__prefix}/lib/rpm/ppc*
%endif
%ifarch s390 s390x
%attr(-, rpm, rpm)          %{__prefix}/lib/rpm/s390*
%endif
%ifarch armv3l armv4l
%attr(-, rpm, rpm)          %{__prefix}/lib/rpm/armv[34][lb]*
%endif
%ifarch mips mipsel
%attr(-, rpm, rpm)          %{__prefix}/lib/rpm/mips*
%endif
%ifarch x86_64
%attr(-, rpm, rpm)          %{__prefix}/lib/rpm/x86_64*
%endif
%attr(-, rpm, rpm)          %{__prefix}/lib/rpm/noarch*

%lang(cs)   %{__prefix}/*/locale/cs/LC_MESSAGES/rpm.mo
%lang(da)   %{__prefix}/*/locale/da/LC_MESSAGES/rpm.mo
%lang(de)   %{__prefix}/*/locale/de/LC_MESSAGES/rpm.mo
%lang(fi)   %{__prefix}/*/locale/fi/LC_MESSAGES/rpm.mo
%lang(fr)   %{__prefix}/*/locale/fr/LC_MESSAGES/rpm.mo
%lang(gl)   %{__prefix}/*/locale/gl/LC_MESSAGES/rpm.mo
%lang(is)   %{__prefix}/*/locale/is/LC_MESSAGES/rpm.mo
%lang(ja)   %{__prefix}/*/locale/ja/LC_MESSAGES/rpm.mo
%lang(ko)   %{__prefix}/*/locale/ko/LC_MESSAGES/rpm.mo
%lang(no)   %{__prefix}/*/locale/no/LC_MESSAGES/rpm.mo
%lang(pl)   %{__prefix}/*/locale/pl/LC_MESSAGES/rpm.mo
%lang(pt)   %{__prefix}/*/locale/pt/LC_MESSAGES/rpm.mo
%lang(pt_BR)      %{__prefix}/*/locale/pt_BR/LC_MESSAGES/rpm.mo
%lang(ro)   %{__prefix}/*/locale/ro/LC_MESSAGES/rpm.mo
%lang(ru)   %{__prefix}/*/locale/ru/LC_MESSAGES/rpm.mo
%lang(sk)   %{__prefix}/*/locale/sk/LC_MESSAGES/rpm.mo
%lang(sl)   %{__prefix}/*/locale/sl/LC_MESSAGES/rpm.mo
%lang(sr)   %{__prefix}/*/locale/sr/LC_MESSAGES/rpm.mo
%lang(sv)   %{__prefix}/*/locale/sv/LC_MESSAGES/rpm.mo
%lang(tr)   %{__prefix}/*/locale/tr/LC_MESSAGES/rpm.mo

%{__mandir}/man1/gendiff.1*
%{__mandir}/man8/rpm.8*
%{__mandir}/man8/rpm2cpio.8*
%lang(ja)   %{__mandir}/ja/man[18]/*.[18]*
%lang(pl)   %{__mandir}/pl/man[18]/*.[18]*
```

```
%lang(ru)   %{__mandir}/ru/man[18]/*.[18]*
%lang(sk)   %{__mandir}/sk/man[18]/*.[18]*

%files build
%defattr(-,root,root)
%dir %{__prefix}/src/winona
%dir %{__prefix}/src/winona/BUILD
%dir %{__prefix}/src/winona/SPECS
%dir %{__prefix}/src/winona/SOURCES
%dir %{__prefix}/src/winona/SRPMS
%dir %{__prefix}/src/winona/RPMS
%{__prefix}/src/winona/RPMS/*
%rpmattr    %{__bindir}/rpmbuild
%rpmattr    %{__prefix}/lib/rpm/brp-*
%rpmattr    %{__prefix}/lib/rpm/check-files
%rpmattr    %{__prefix}/lib/rpm/check-prereqs
%rpmattr    %{__prefix}/lib/rpm/config.site
%rpmattr    %{__prefix}/lib/rpm/cross-build
%rpmattr    %{__prefix}/lib/rpm/debugedit
%rpmattr    %{__prefix}/lib/rpm/find-debuginfo.sh
%rpmattr    %{__prefix}/lib/rpm/find-lang.sh
%rpmattr    %{__prefix}/lib/rpm/find-prov.pl
%rpmattr    %{__prefix}/lib/rpm/find-provides
%rpmattr    %{__prefix}/lib/rpm/find-provides.perl
%rpmattr    %{__prefix}/lib/rpm/find-req.pl
%rpmattr    %{__prefix}/lib/rpm/find-requires
%rpmattr    %{__prefix}/lib/rpm/find-requires.perl
%rpmattr    %{__prefix}/lib/rpm/get_magic.pl
%rpmattr    %{__prefix}/lib/rpm/getpo.sh
%rpmattr    %{__prefix}/lib/rpm/http.req
%rpmattr    %{__prefix}/lib/rpm/javadeps
%rpmattr    %{__prefix}/lib/rpm/magic
%rpmattr    %{__prefix}/lib/rpm/magic.mgc
%rpmattr    %{__prefix}/lib/rpm/magic.mime
%rpmattr    %{__prefix}/lib/rpm/magic.mime.mgc
%rpmattr    %{__prefix}/lib/rpm/magic.prov
%rpmattr    %{__prefix}/lib/rpm/magic.req
%rpmattr    %{__prefix}/lib/rpm/perldeps.pl
%rpmattr    %{__prefix}/lib/rpm/perl.prov
%rpmattr    %{__prefix}/lib/rpm/perl.req

%rpmattr    %{__prefix}/lib/rpm/rpm[bt]
%rpmattr    %{__prefix}/lib/rpm/rpmdeps
%rpmattr    %{__prefix}/lib/rpm/trpm
%rpmattr    %{__prefix}/lib/rpm/u_pkg.sh
%rpmattr    %{__prefix}/lib/rpm/vpkg-provides.sh
%rpmattr    %{__prefix}/lib/rpm/vpkg-provides2.sh

%{__mandir}/man8/rpmbuild.8*
%{__mandir}/man8/rpmdeps.8*

%if %{with_python_subpackage}
%files python
%defattr(-,root,root)
%{__libdir}/python%{with_python_version}/site-packages/rpmmodule.so
%{__libdir}/python%{with_python_version}/site-packages/rpmdb
%endif

%files devel
%defattr(-,root,root)
%if %{with_apidocs}
%doc apidocs
%endif
%{__includedir}/rpm
%{__libdir}/librpm.a
%{__libdir}/librpm.la
%{__libdir}/librpm.so
%{__libdir}/librpmdb.a
```

```
%{__libdir}/librpmdb.la
%{__libdir}/librpmdb.so
%{__libdir}/librpmio.a
%{__libdir}/librpmio.la
%{__libdir}/librpmio.so
%{__libdir}/librpmbuild.a
%{__libdir}/librpmbuild.la
%{__libdir}/librpmbuild.so
%{__mandir}/man8/rpmcache.8*
%{__mandir}/man8/rpmgraph.8*
%rpmattr    %{__prefix}/lib/rpm/rpmcache
%rpmattr    %{__prefix}/lib/rpm/rpmdb_deadlock
%rpmattr    %{__prefix}/lib/rpm/rpmdb_dump
%rpmattr    %{__prefix}/lib/rpm/rpmdb_load
%rpmattr    %{__prefix}/lib/rpm/rpmdb_loadcvt
%rpmattr    %{__prefix}/lib/rpm/rpmdb_svc
%rpmattr    %{__prefix}/lib/rpm/rpmdb_stat
%rpmattr    %{__prefix}/lib/rpm/rpmdb_verify
%rpmattr    %{__prefix}/lib/rpm/rpmfile
%rpmattr    %{__bindir}/rpmgraph

%files -n popt
%defattr(-,root,root)
%{__libdir}/libpopt.so.*
%{__mandir}/man3/popt.3*
%lang(cs)   %{__prefix}/*/locale/cs/LC_MESSAGES/popt.mo
%lang(da)   %{__prefix}/*/locale/da/LC_MESSAGES/popt.mo
%lang(de)   %{__prefix}/*/locale/de/LC_MESSAGES/popt.mo
%lang(es)   %{__prefix}/*/locale/es/LC_MESSAGES/popt.mo
%lang(eu_ES)      %{__prefix}/*/locale/eu_ES/LC_MESSAGES/popt.mo
%lang(fi)   %{__prefix}/*/locale/fi/LC_MESSAGES/popt.mo
%lang(fr)   %{__prefix}/*/locale/fr/LC_MESSAGES/popt.mo
%lang(gl)   %{__prefix}/*/locale/gl/LC_MESSAGES/popt.mo
%lang(hu)   %{__prefix}/*/locale/hu/LC_MESSAGES/popt.mo
%lang(id)   %{__prefix}/*/locale/id/LC_MESSAGES/popt.mo
%lang(is)   %{__prefix}/*/locale/is/LC_MESSAGES/popt.mo
%lang(it)   %{__prefix}/*/locale/it/LC_MESSAGES/popt.mo
%lang(ja)   %{__prefix}/*/locale/ja/LC_MESSAGES/popt.mo
%lang(ko)   %{__prefix}/*/locale/ko/LC_MESSAGES/popt.mo
%lang(no)   %{__prefix}/*/locale/no/LC_MESSAGES/popt.mo
%lang(pl)   %{__prefix}/*/locale/pl/LC_MESSAGES/popt.mo
%lang(pt)   %{__prefix}/*/locale/pt/LC_MESSAGES/popt.mo
%lang(pt_BR)      %{__prefix}/*/locale/pt_BR/LC_MESSAGES/popt.mo
%lang(ro)   %{__prefix}/*/locale/ro/LC_MESSAGES/popt.mo
%lang(ru)   %{__prefix}/*/locale/ru/LC_MESSAGES/popt.mo
%lang(sk)   %{__prefix}/*/locale/sk/LC_MESSAGES/popt.mo
%lang(sl)   %{__prefix}/*/locale/sl/LC_MESSAGES/popt.mo
%lang(sr)   %{__prefix}/*/locale/sr/LC_MESSAGES/popt.mo
%lang(sv)   %{__prefix}/*/locale/sv/LC_MESSAGES/popt.mo
%lang(tr)   %{__prefix}/*/locale/tr/LC_MESSAGES/popt.mo
%lang(uk)   %{__prefix}/*/locale/uk/LC_MESSAGES/popt.mo
%lang(wa)   %{__prefix}/*/locale/wa/LC_MESSAGES/popt.mo
%lang(zh)   %{__prefix}/*/locale/zh/LC_MESSAGES/popt.mo
%lang(zh_CN)      %{__prefix}/*/locale/zh_CN.GB2312/LC_MESSAGES/popt.mo

# XXX These may end up in popt-devel but it hardly seems worth the effort.
%{__libdir}/libpopt.a
%{__libdir}/libpopt.la
%{__libdir}/libpopt.so
%{__includedir}/popt.h

%changelog
* Mon Feb 02 2004 Ulisses Muniz de Queiroz <ulissesq@terra.com.br>
- macros now suites Winona
- add elfutils-devel as pre-req
```

8.6.7 anaconda-images.spec

```
Summary: Images used in the Red Hat Linux installer
Name: anaconda-images
Version: 9.2
Release: 4
Source0: %{name}-%{version}.tar.bz2
License: Copyright © 2002 Red Hat, Inc.  All rights reserved.
Group: Applications/System
BuildRoot: %{_tmppath}/%{name}-root
BuildArch: noarch
Requires: anaconda-runtime

%description
The anaconda-images package (the "Package") contain image files which
incorporate the Fedora trademark and the RPM logo (the "Marks").  The
Marks are trademarks or registered trademarks of Red Hat, Inc. in the
United States and other countries and are used by permission.

Please see the included COPYING file for information on copying
and redistribution.

%prep
%setup -q

%build
make

%install
rm -rf $RPM_BUILD_ROOT

make DESTDIR=$RPM_BUILD_ROOT install

%clean
rm -rf $RPM_BUILD_ROOT

%files
%defattr(-,root,root)
%doc COPYING
%dir %{_datadir}/anaconda
%{_datadir}/anaconda/pixmaps
%{_prefix}/lib/anaconda-runtime/boot/syslinux-splash.png

%changelog
* Fri Jan 30 2004 Crineu Tres <crineu@inf.ufsc.br>
- replaced default artwork with Winona's artwork
```

8.6.8 flash-player.spec

```
%define ver 6.0
%define rel 79
%define topdir install_flash_player_6_linux
%define libstdc libstdc++-libc6.2-2.so.3
%define mozilladir $(rpmquery --list mozilla | grep libnullplugin.so | sed -e
's;plugins/libnullplugin.so;;'g)

Name: flash-player
Version: %{ver}.%{rel}
Release: 1
Summary: Macromedia Flash Player.
Group: Applications/Internet
Group(pt_BR): AplicaÃ§Ãµes/Internet
License: MACROMEDIA SOFTWARE END USER LICENSE
URL: http://www.macromedia.com/software/flashplayer/
Source:
http://download.macromedia.com/pub/shockwave/flash/english/linux/%{ver}r%{rel}/%{to
pdir}.tar.gz
Requires: libstdc++ mozilla
```

```
Provides: %{libstdc}
BuildRoot: %{_tmppath}/%{name}-%{version}-root

%description
Macromedia Flash lets designers and developers integrate video, text, audio, and
graphics into effective experiences. Macromedia Flash is the worldâs most
pervasive software platform, reaching 98% of Internet-enabled desktops worldwide,
as well as many popular devices.

%description -l pt_BR
O Macromedia Flash permite que projetistas e desenvolvedores integrem vÃ-deo,
texto, Ã¡udio e grÃ¡ficos em experiÃªncias eficientes. Macromedia Flash Ã© a
plataforma de software mais difundida do mundo, chegando Ã  98% dos computadores
conectados Ã  Internet no mundo todo, assim como Ã  muitos dispositivos populares.

%prep
%setup -q -n install_flash_player_6_linux

%install
rm -rf %{buildroot}
MOZILLADIR=%{mozilladir}
%{__mkdir_p} %{buildroot}%{_libdir}/mozilla/plugins/
%{__mkdir_p} %{buildroot}$MOZILLADIR
%{__ln_s}f %{_libdir}/libstdc++.so.5 %{buildroot}${MOZILLADIR}%{libstdc}
cp flashplayer.xpt libflashplayer.so %{buildroot}%{_libdir}/mozilla/plugins/

%clean
rm -rf %{buildroot}

%files
%defattr(755,root,root)
%{_libdir}/mozilla-*/*
%{_libdir}/mozilla/plugins/*
```

8.6.9 redhat-menus.spec

```
%define gettext_package redhat-menus

Summary: Configuration and data files for the desktop menus
Name: redhat-menus
Version: 0.40
Release: 2
URL: http://www.redhat.com
Source0: %{name}-%{version}.tar.gz
Patch: %{name}-0.40-pt_BR.patch
License: XFree86
Group: User Interface/Desktops
BuildRoot: %{_tmppath}/%{name}-root
BuildArchitectures: noarch

## old nautilus contained start-here stuff
Conflicts: nautilus <= 2.0.3-1
## desktop files in redhat-menus point to icons in new artwork
Conflicts: redhat-artwork < 0.35

%description

This package contains the XML files that describe the menu layout for
GNOME and KDE, and the .desktop files that define the names and icons
of "subdirectories" in the menus.

%prep
%setup -q
%patch -p 1

%build
```

```
%configure
make

# the "perl only English" trick was OK for translations, but
# still broke docs - so can't do this
#perl -pi -e 's/Web Browser/Mozilla Web Browser/g' desktop-files/redhat-web.desktop
#perl -pi -e 's/Email/Evolution Email/g' desktop-files/redhat-email.desktop
#perl -pi -e 's/Diagrams/Dia Diagrams/g' desktop-files/redhat-diagrams.desktop
#perl -pi -e 's/Video Conferencing/GnomeMeeting Video Conferencing/g' desktop-
files/redhat-gnomemeeting.desktop

%install
rm -rf $RPM_BUILD_ROOT

%makeinstall

%find_lang %{gettext_package}

%clean
rm -rf $RPM_BUILD_ROOT

%files  -f %{gettext_package}.lang
%defattr(-,root,root)
%dir %{_sysconfdir}/X11/desktop-menus
%config %{_sysconfdir}/X11/desktop-menus/*
%{_sysconfdir}/X11/starthere
%{_datadir}/desktop-menu-files
%{_datadir}/desktop-menu-patches
%{_datadir}/applications

%changelog
* Tue Feb 3 2004 Ulisses Muniz de Queiroz <ulissesq@terra.com.br> 0.40-2
- fix pt_BR strings
```

8.6.10 redhat-menus-0.40-pt_BR.patch

```
--- redhat-menus-0.40/po/pt_BR.po.orig  2004-02-03 21:08:34.000000000 -0200
+++ redhat-menus-0.40/po/pt_BR.po  2004-02-03 21:15:02.000000000 -0200
@@ -176,19 +176,19 @@

 #: desktop-files/Office.directory.in.h:1
 msgid "Office"
-msgstr "Office"
+msgstr "Escritório"

 #: desktop-files/Office.directory.in.h:2
 msgid "Office Applications"
-msgstr "Aplicações do Office"
+msgstr "Aplicações de Escritório"

 #: desktop-files/Office-More.directory.in.h:1
 msgid "Additional office applications"
-msgstr "Aplicações adicionais do Office"
+msgstr "Aplicações adicionais de escritório"

 #: desktop-files/Office-More.directory.in.h:2
 msgid "More Office Applications"
-msgstr "Mais Aplicações do Office"
+msgstr "Mais Aplicações de Escritório"

 #: desktop-files/openoffice-printeradmin.desktop.in.h:1
 msgid "OpenOffice.org Printer Setup"
```