fisl14

14º Fórum Internacional

Software Livre

A tecnologia que liberta

Seja bem-vindo!

O Fórum Internacional Software Livre (FISL) é um dos maiores e mais importantes eventos de tecnologia da América Latina. É realizado há 14 anos em Porto Alegre e, na última edição, reuniu cerca de 8 mil pessoas em quatro dias, contando com mais de 800 horas de palestras e cerca de 600 palestrantes.

Apesar de toda esta visibilidade, ainda é comum que pessoas que não fazem parte desta comunidade tenham dúvidas e achem o assunto um tanto quanto hermético. Por mais complexo que possa parecer, na verdade, falar sobre software livre é falar sobre liberdade. Por esse motivo, preparamos este workshop visando compartilhar conhecimento com os jornalistas e esclarecendo as principais dúvidas sobre o tema.

Neste material reunimos dois textos que apresentam a história do movimento software livre e os principais conceitos por trás de toda esta filosofia. Esperamos que aproveitem a experiência e contamos com a sua presença no fisl14, de 3 a 6 de julho, no Centro de Eventos da PUCRS.

GT-Comunicação
Fórum Internacional Software Livre – FISL

Para mergulhar no Software Livre

Revista Select Mariel Zasso

Na sua 13ª edição, o Fórum Internacional Software Livre – fisl, é o mais significativo encontro de comunidades de software livre na América Latina.

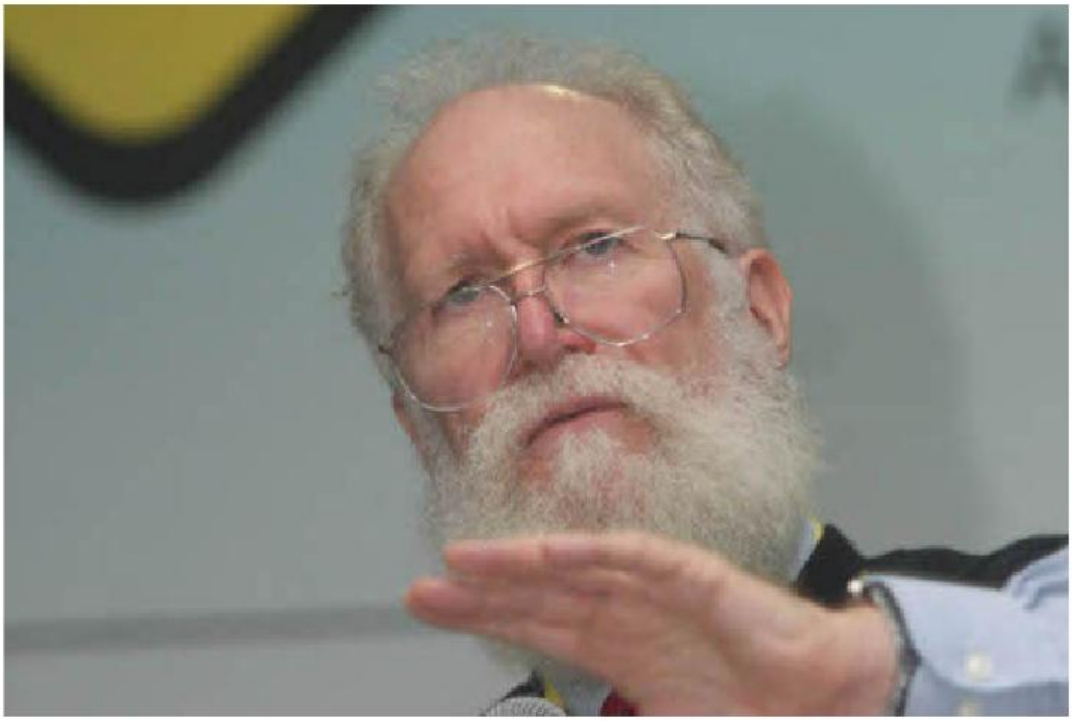
Da motivação de um grupo de pesquisadores e ativistas pelo conhecimento compartilhado nasceu o que viria a se tornar o maior evento de Software Livre da América Latina. A primeira edição do Fórum Internacional Software Livre foi realizada no ano 2000, e surpreendeu os organizadores pelo interesse e participação.

Treze anos depois, o evento tornou-se um marco no calendário de interessados em tecnologias livres no mundo todo, tendo sediado inclusive encontros inusitados como o do então presidente Lula com o Peter Sunde, um dos três suecos que fundaram o The Pirate Bay, que na época acabara de ser acusado judicialmente por ajudar milhões de internautas a quebrar leis de direitos autorais.

A edição deste ano do fisl segue a tradição de evento montado comunitariamente como funciona a maioria dos projetos de softwares livres, desenvolvidos, muitas vezes de modo voluntário, por comunidades de programadores e tem sua grade de atividades proposta e curada pelos participantes. Qualquer pessoa pode propor uma atividade dentro das temáticas propostas pelo Fórum, e após encerrado o período de inscrições de palestrantes, estes são convidados a votar e escolher suas favoritas.

O jovem sueco fundador do Pirate Bay era perseguido no seu país quando encontrou Lula durante a 10ª edição do fisl, em 2009. (Foto: Mariel Zasso)

Além das propostas gerais, um comitê de programação convida palestrantes de renome internacional em suas áreas para complementar o time. Um dos responsáveis pelo nascimento da moeda peer-to-peer BitCoin , Amir Taaki, dois dos principais desenvolvedores do player VLC, Jean-Baptiste Kempf e Felix Paul Kuhne, além do presidente-fundador da Linux Foundation, um dos ídolos das comunidades de Software Livre, John Maddog Hall, são só alguns dos grandes nomes que passarão por lá.



Para John Maddog Hall, o fisl é compromisso anual marcado na agenda. Não há participante do evento que não tenha uma foto a seu lado.
(Foto: Cristiano Sant'Anna/divulgação)

Engana-se, porém, quem pensar que o fisl é um evento voltado apenas para desenvolvedores de software: a programação contempla temas como ecossistema, cultura digital, robótica e metareciclagem, educação e negócios. Todos eles explorados em suas relações com o software livre.

Mas o que é afinal software livre?

A primeira distinção a qual um leigo costuma ser submetido é: software livre não é software grátis. Um software pago pode ser livre, e vice-versa. Em parte, a confusão é um resquício do idioma de origem do termo free software. Em inglês, a mesma palavra free serve tanto para "livre", como em liberdade, como pra "grátis", como em "free beer" (cerveja grátis).

Mas o que está em questão aqui não é uma distribuição gratuita de programas de computador. Um software é considerado livre quando seus criadores explicitam na sua licença as "quatro liberdades": liberdade para usá-lo para qualquer propósito, liberdade para estudar seu funcionamento e adaptá-lo às suas necessidades, liberdade para fazer cópias e distribui-lo (mesmo que cobrando por isso), e liberdade para aperfeiçoá-lo.

Para estudar o funcionamento e para poder aperfeiçoar ou adaptar um software, é preciso ter acesso à "receita" de como ele é feito: o código-fonte. Todo software livre tem obrigatoriamente seu código-fonte público, acessível a qualquer interessado, que será livre para modificar e inclusive inventar um novo programa sobre as mesmas bases se quiser. Para a maioria de nós, não-programadores e pouco ou nada aptos a fazer tais modificações, isso pode parecer sem importância. Mas vejamos agora porque isso diz respeito também a você.

No início, todo software era livre

Considerado o fundador do movimento, Richard Stallman veio a Porto Alegre no início de junho falar sobre software livre e lançar oficialmente a 13ª edição do evento. (Foto: Cristiano Sant'Anna/divulgação)

As chamadas "quatro liberdades" fazem parte da definição da Free Software Foundation, fundada em 1985 por Richard Stallman com o objetivo de promover a adoção do software livre, em outras palavras, promover a eliminação de restrições sobre a cópia, redistribuição, estudo e modificação de programas de computadores bandeiras do movimento.

É certo que toda ideia nasce de uma necessidade, e não foi diferente neste caso. Depois de trabalhar como programador em projetos de software no MIT, Stallman revoltou-se com os rumos que a pesquisa e a indústria de softwares estavam tomando, e em 1984, afasta-se da universidade para criar o projeto GNU, uma opção deliberada de rejeitar a noção de software como segredo industrial.

Até pouco tempo antes, o conceito de software livre não existia simplesmente porque todos os softwares o eram. Ninguém pensara até então em tratar códigos-fonte como segredos. Mas a ainda recente indústria de programas para computador logo vislumbrou o mercado crescente. E a revolta de Stallman, estopim do movimento, surge de uma situação muito prosaica.

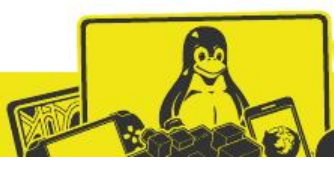
Uma impressora que entrou para a história

Nos anos 80, no MIT, a velha impressora coletiva apresentava problemas, e os programadores não se furtaram a melhorar o software que a comandava para resolver inoperâncias do dia-a-dia. Mas eis que chega uma impressora nova, um presente caro e moderno cortesia do próprio fabricante. A nova máquina também apresenta problemas, os rapazes acreditam que são capazes de resolvê-los da maneira que já havia funcionado. Mas o software da impressora não incluia acesso a seu código-fonte. Não apenas o fabricante não oferecera, mas também o colega que teria trabalhado nele negou-se a compartilhar: tinha assinado um contrato que exigia segredo.

Com todas as produtoras de software encaminhando-se para rumos semelhantes, parecia haver apenas uma saída para continuar sendo programador: assinar um contrato no qual prometesse não compartilhar o código-fonte, e melhorar os softwares como empregado da empresa.

Mas Richard Stallman criou a via do meio: abandonou o MIT, fundou a Free Software Foundation, e continua lutando para manter livre o fluxo de ideias na área de softwares. O movimento software livre inspira ainda hoje, direta ou indiretamente, diversas vertentes de luta pelo conhecimento como bem comum, em vez de segredo de corporações. Copyleft, creative commons, uma verdadeira revolução nas noções modernas de direitos autorais, brotam também da internet e são certamente fruto do espírito do tempo. Mas o movimento sofware livre, na figura de Richard Stallman, merece os louros por ter gritado o primeiro basta.

Publicado originalmente no portal da Revista Select em 25 de junho de 2012.



Descobrindo o Linux

Adaptado do livro de João Eriberto Mota Filho

Índice

1-Definição de Software Livre

A expressão "Software Livre" gera uma enorme confusão na cabeça das pessoas. Muitos pensam que Software Livre (ou "free software") é algo gratuito. O termo "free" está ligado a livre e não a gratuito.

Software livre é um conceito especial. Esse conceito prevê que todo software será distribuído com seu código-fonte, podendo ser alterado e até mesmo redistribuído depois de alterado. Mas esse software não precisa ser gratuito. O seu pagamento pode se dar de várias formas. Por exemplo: você produz um banco de dados e o vende por uma determinada quantia. Isso irá custear a mídia, a embalagem etc. Quem quiser, poderá copiar livremente ou alterar o código e não terá de lhe pagar nada. No entanto, você pode cobrar pelo suporte técnico. Pode ser um contrato mensal, por exemplo. É daí que vem o lucro. Outro exemplo: você faz um software e o vende dentro de uma caixa que também contém um manual com mil páginas. Qualquer pessoa pode adquirir uma cópia, gratuitamente, de outra pessoa que já tenha comprado o pacote. No entanto, não haverá o manual. Muitos irão preferir ter a bela caixa e o manual, comprando diretamente de você. Mas não se esqueça: em qualquer um desses casos, sempre haverá o código-fonte e, se uma pessoa passar uma cópia para outra, não será um caso de pirataria.

Segundo a definição de Richard Stallman, o Software Livre nos proporciona:

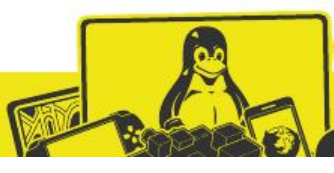
• a liberdade de executar um programa, seja qual for o propósito;

• a liberdade de modificar um programa para adaptá-lo às suas necessidades e, para que isso ocorra, você deve ter acesso ao código-fonte;

• a liberdade de redistribuir cópias, gratuitamente ou mediante uma taxa;

• a liberdade de distribuir versões modificadas do programa e, nesse caso, toda a comunidade poderá beneficiar-se dos aperfeiçoamentos.

Temos de saber diferenciar free software e freeware. O free software, na sua mais ampla concepção, traz consigo o código-fonte, pode ser vendido e ser livremente alterado, adaptado e redistribuído. O freeware é obrigatoriamente de graça, mas não traz consigo o código-fonte e, em consequência, não pode ser alterado.



2-Free software e open source

Dois termos muito utilizados atualmente são free software (ou Software Livre) e open source (ou código aberto). Muitos encontram dificuldades em explicar a diferença entre os termos. A grande verdade é que free software e open source são a mesma coisa. Software livre, já definido anteriormente, refere-se à liberdade de poder executar, estudar, modificar e redistribuir versões, originais ou modificadas, de um programa. Assim sendo, o Software Livre é uma filosofia, uma forma de pensar. Open source seria um modelo de desenvolvimento que, no fim, respeita os mesmos princípios do Software Livre.

A Open Source Initiative, cujo site é http://www.opensource.org, estabeleceu um conceito de open source baseado na Definição Debian de Software Livre (DFSG – Debian Free Software Guidelines), disponível em http://www.debian.org/social_contract. Segundo a Open Source Initiative, um open source deve seguir preceitos referentes aos seguintes tópicos:

- redistribuição livre;
- código-fonte;
- trabalhos derivados;
- integridade do código-fonte do autor;
- não-discriminação às pessoas ou grupos;
- não-discriminação às diversas intenções de utilização;
- a licença não deve ser específica para um produto;
- a licença não deve restringir outro software;
- a licença não pode ser calcada sobre qualquer tecnologia.

O maior defensor do conceito "Open Source" é Eric Reymond. Eric descreve o modelo de desenvolvimento open source no livro "The Cathedral and the Bazaar" (A Catedral e o Bazar).

Alguns fornecedores de software distorcem um pouco o conceito de código aberto, disponibilizando o código-fonte de um programa, total ou parcialmente, mediante uma licença que restringe o uso desse código de alguma forma. Recentemente, surgiu o conceito de FOSS (Free and Open Source Software). Esse conceito foi criado, principalmente, para mesclar comunidades em eventos, simpósios, seminários, trabalhos conjuntos etc.

3-Licença GNU GPL

A Licença GNU General Public License foi desenvolvida pela Free Software Foundation (FSF) para especificar se um software é livre ou não. Existem várias outras licenças, inclusive compatíveis com a GNU GPL, mas essa é a mais recomendada. Numa avaliação geral, a GNU GPL se baseia nas quatro liberdades básicas: executar, estudar, modificar e redistribuir versões, originais ou modificadas, de um programa. A Licença GNU GPL está disponível em http://www.gnu.org/copyleft/gpl.html.

É importante ressaltar que uma licença é um acordo entre partes. Nesse caso, entre o desenvolvedor e os usuários finais. Assim sendo, geralmente, uma licença como a GNU GPL tem validade no Brasil e em outros países. Esse fato esgota dúvidas, tais como: a GPL tem validade no Brasil? A resposta é sim, mesmo que não na sua totalidade (uma licença poderá conter cláusulas consideradas abusivas, por exemplo, em um determinado país).

4-O sistema operacional GNU/Linux

Mas o que é o Linux afinal? Essa pergunta também causa confusão a muitas pessoas. Linus, em seu e-mail inicial, disse o seguinte: “estou fazendo um sistema operacional” .

A Free Software Foundation começou o GNU pelos aplicativos e ainda não conseguiu terminar o kernel (Hurd). Linus, ao contrário da FSF, começou pelo kernel. No entanto, nunca chegou a desenvolver os aplicativos. Ou seja: o Linux é só um kernel, não um sistema operacional. E nos tempos atuais, não há mais a intenção de fazer um sistema operacional completo. Manter um kernel, mesmo trabalhando em comunidade, já é trabalhoso o suficiente.

Linus utilizou exaustivamente os programas gerados pela FSF para o Projeto GNU. A lógica era simples: os programas da FSF funcionavam no Unix. Se funcionassem também no Linux, seria um sinal de que o kernel estaria ajustado e similar ao kernel do Unix. Desde o início, Linus distribuiu o seu kernel com programas da FSF, promovendo uma integração (um pouco unilateral, diga-se de passagem).

Por tudo isso, o nome Linux refere-se apenas ao kernel criado por Linus Torvalds. O sistema operacional que conhecemos por Linux chama-se, na verdade, sistema operacional GNU/Linux. Essa é a forma correta de se referir ao conjunto, ao sistema operacional.

Normalmente, falamos apenas Linux. Isso ocorre tanto pelo desconhecimento quanto pela comodidade. No entanto, neste livro, por uma questão didática, procuraremos sempre utilizar o nome GNU/Linux para o sistema operacional e somente Linux para o kernel.

Cabe ainda ressaltar que o Kernel Linux, desde o início, contou com a ajuda de vários desenvolvedores para tornar-se o que é hoje. Alan Cox é um profundo conhecedor do kernel e ajuda Linus nas partes mais difíceis do desenvolvimento. Mas lembre-se: o Linux é desenvolvido por pessoas do mundo todo, da comunidade livre.

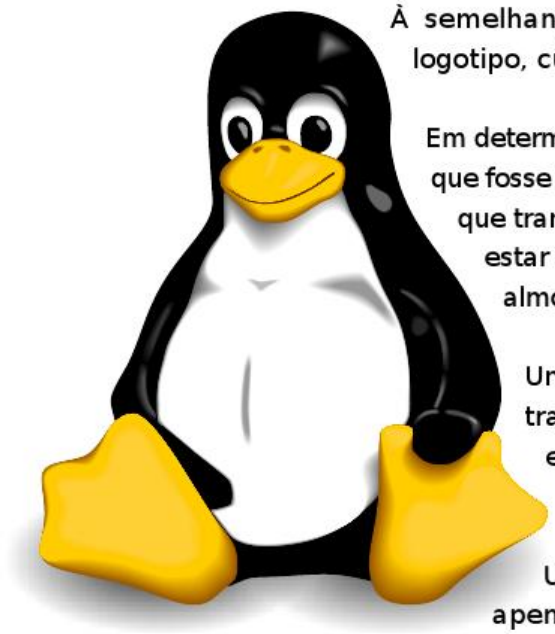
5-Motivos para criar o Linux

Em uma entrevista foi perguntado a Linus: "O que o levou a escrever o Linux?" A resposta foi a seguinte: "Bem, como eu disse, queria um determinado desempenho em casa e o DOS (e o Windows) não me ofereciam isso. Comecei tentando um pequeno clone do Unix, chamado Minix. Eu era capaz de entender algo sobre as coisas que pretendia com ele. Por outro lado, faltava-me a plena funcionalidade do Unix. A simplicidade do Minix (e os problemas de performance do Minix) levaram-me a desejar algo melhor. No entanto, o Unix custava muito e não seria fácil encontrar algo bom sem dinheiro (que eu definitivamente não tinha). Uma versão de Unix razoavelmente boa, com ferramentas de desenvolvimento etc, custava alguns milhares de dólares. Como eu era um estudante pobre e havia usado todo o meu dinheiro para comprar um computador, eu realmente não tinha opção... Mas, como eu conhecia computadores, comecei a fazer um sistema para mim mesmo, e o resto da história todos conhecem".

6-Distribuições GNU/Linux

Quando juntamos um Kernel Linux, diversos aplicativos, compiladores etc., alguns da Free Software Foundation (projeto GNU) e outros não, temos uma distribuição GNU/ Linux. Em síntese, distribuição é a união do kernel com vários programas compatíveis com ele. Como no início a esmagadora maioria dos aplicativos utilizados era proveniente do GNU, criou-se o sistema operacional GNU/Linux. Atualmente, existem várias distribuições. O site DistroWatch.com afirma que temos 318 distribuições ativas (dados de março de 2012). As maiores e mais antigas ainda em produção são, na ordem: Slackware, Debian, openSUSE e Red Hat. Muitas das outras distribuições são derivadas dessas.

7-Tux



À semelhança do gnu do Projeto GNU, o Kernel Linux possui um logotipo, cujo nome é Tux (fusão de Torvalds com Unix).

Em determinado momento, Linus Torvalds entrou na conversa e sugeriu que fosse um pinguim um pouco gordo, mas não extremamente obeso, que transmitisse uma imagem de tranquilidade e satisfação. Deveria estar sentado, sorridente e satisfeito, digerindo com felicidade um almoço, o que não o deixaria com vontade de estar em pé.

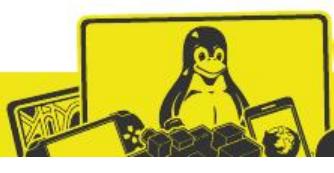
Um longo caminho histórico, repleto de antecedentes, foi traçado até chegarmos ao Kernel Linux. É muito importante entender todo esse caminho. Muitos fatos que conhecemos hoje em dia foram causados por episódios antigos.

Uma consideração importante é o fato de que o Linux é apenas um kernel. Ele foi criado por Linus Torvalds, mas os aplicativos utilizados, desde o começo, foram feitos pela FSF para o projeto GNU. Assim sendo, o nome correto do sistema operacional é GNU/Linux. Erroneamente, ou por falta de conhecimento ou por comodismo, as pessoas utilizam o termo Linux para referenciar algo bem mais amplo do que o Linux realmente é. Mas que fique bem claro: Linux é somente o kernel. E o Tux é o símbolo do kernel, somente.

8-História do GNU/Linux

Todo computador precisa de um sistema operacional para funcionar. O sistema operacional é responsável por controlar a utilização dos recursos fornecidos pela máquina, como processador, memória e discos. Para entender a história do sistema operacional GNU/Linux, será necessário conhecer vários fatos anteriores à sua criação.

O CTSS (Compatible Time-Sharing System) foi um dos primeiros sistemas operacionais a adotar a técnica de time sharing. Essa técnica, empregada até hoje, permite que vários usuários possam, simultaneamente, utilizar um ambiente para executar programas. Tudo isso ocorre sobre o mesmo sistema operacional, rodando em uma máquina. Esse tipo de sistema caracteriza o processo de compartilhamento de processador, memória e disco entre vários utilizadores. O conceito era simples. Fatias de tempo do processador, conhecidas como time slice, seriam destinadas aos programas carregados em memória. Assim, cada programa receberia a atenção individual da máquina por algumas frações de segundo, o que nos dá, até hoje, a ideia de que tudo está funcionando ao mesmo tempo.



Ainda em novembro de 1962, Joseph Carl Robnett Licklider, integrante do MIT, propôs o Projeto MAC, criado para desenvolver dois produtos finais: um sistema operacional avançado e um laboratório de inteligência artificial. Em virtude disso, a sigla do projeto foi tratada com dois nomes diferentes: Multiple Access Computers e Man And Computers. O subprojeto Multiple Access Computers tentaria desenvolver o sistema operacional Multics (MULTiplexed Information and Computing Service). O Multics deveria ser algo superior ao CTSS.

O objetivo final, em relação ao Multics, era um sistema operacional com suporte para memória virtual, utilizando recursos de paginação e segmentação de memória. Isso possibilitaria um sofisticado processo de transferência de dados entre discos e memória.

Após isso, surgiu o Unics, uma tentativa de fazer, com rapidez, um sistema operacional simples, versátil e moderno, mantendo-se as ideias de time sharing e de portabilidade entre computadores de todos os tamanhos. O nome surgiu como um trocadilho em relação ao Multics, uma vez que o Unics seria um Multics modesto. Algum tempo depois, em 1970, o nome foi mudado de Unics para Unix.

A primeira versão do Unix foi escrita em Assembly, uma complicada linguagem de baixo nível. Thompson tinha a intenção de passar o Unix para uma linguagem de alto nível. Com o surgimento da linguagem C, o Unix precisou ser reescrito e isso significava começar tudo de novo. Foi um processo lento, iniciado em meados de 1972.

O Unix se espalhou rapidamente pelo mundo acadêmico. Não havia dúvidas de que o mesmo poderia ser uma excepcional fonte de renda. A primeira ideia foi desenvolver programas para Unix para uso comercial. O principal diferencial do Unix era o sistema de time sharing, que permitia às pessoas compartilharem o mesmo computador ao mesmo tempo, utilizando os seus vários terminais. A portabilidade entre máquinas era grande. Os usuários da máquina poderiam trocar e-mails.

Várias versões de Unix foram produzidas. Muitas empresas passaram a vender máquinas projetadas para o uso com o Unix, dentre elas a Sun Microsystems, a SGI, a Hewlett-Packard, a NCR e a IBM. Em paralelo, na Universidade de Berkeley, um trabalho constante introduziu melhorias e versatilidade, criando outra importante versão, também muito utilizada, a BSD. BSD é a abreviatura de Berkeley Software Distribution, uma linha Unix desenvolvida em Berkeley que visa um produto final gratuito. Os BSD e derivados são famosos e muito utilizados. Os mais conhecidos são o MacOS X, o FreeBSD, o OpenBSD e o NetBSD.

9-Richard Stallman

Desde 1971, Richard Matthew Stallman trabalhava no Laboratório de Inteligência Artificial do MIT, desenvolvendo em uma máquina PDP-10 que rodava um sistema operacional chamado ITS (Incompatible Timesharing System, um trocadilho para o CTTS). Esse sistema operacional foi desenvolvido pelos próprios funcionários do laboratório. Ele não era comercial e os desenvolvedores do MIT o aperfeiçoavam cada vez mais. Stallman também fazia parte de uma comunidade voltada para o compartilhamento e a distribuição de software. Essa comunidade existiu por alguns anos. Naquela época ainda não havia o termo "free software" ou, como conhecemos, Software Livre. Mas, apesar do nome não existir ainda, o conceito de Software Livre já era aplicado. Segundo Stallman, "Quando alguém de outra universidade ou empresa precisava usar um programa do Laboratório de Inteligência, nós deixávamos com satisfação. E se você visse alguém usando um programa desconhecido e interessante, poderia pedir para ver o código dele também. Com isso, você poderia ler o código, alterá-lo e até aproveitar partes dele para gerar um novo programa"

Toda essa situação de liberdade mudou drasticamente no início da década de 1980, quando a Digital descontinuou o PDP-10. A comunidade, liderada por desenvolvedores que trabalhavam com o PDP-10 no MIT, começou a desmanchar-se. O MIT resolveu comprar uma nova máquina, substituta do PDP-10, e um novo sistema operacional. O ITS, utilizado até então no PDP-10, não era baseado em time sharing e, portanto, era incompatível com os novos tempos. Assim, em 1982, a ideia dos administradores do MIT era comprar um software, não livre, baseado em time sharing. Os computadores modernos, como o VAX da Digital, tinham o seu próprio sistema operacional e nada mais era livre. Com isso, tornava-se necessário assinar um termo de confidencialidade para receber uma cópia do executável. Apenas o executável, nada de código-fonte. Nas próprias palavras de Stallman, "isso significava prometer não ajudar a quem precisasse; era uma proibição de uma comunidade colaborativa" As regras do contrato diziam: "se compartilhar o software com alguém, você será um pirata" .

Ainda: "se precisar de alguma alteração no software, peça-nos para fazê-la para você" . Stallman já havia tido problemas com termos de confidencialidade em software proprietário, ainda na década de 1970. Naquela ocasião, o Laboratório de Inteligência Artificial havia recebido uma impressora laser, da marca Xerox. Ao

que consta, era a única laser da Xerox, no mundo, que não se encontrava dentro da própria Xerox. Na verdade, era uma adaptação de uma copiadora. Essa impressora estava acoplada a um PDP-10 e, constantemente, apresentava problemas com o papel, que prendia no rolo pressor. Esse problema só era detectado quando se estava diante da impressora, o que causava uma perda de tempo enorme porque, mediante uma falha, não se podia adotar uma ação imediata. A solução seria alterar o driver da impressora para que a mesma pudesse avisar sobre a ocorrência de falhas. Quase tudo naquela época era livre, exceto o driver daquela impressora, que envolvia um termo de confidencialidade, uma vez que a mesma era utilizada somente dentro da Xerox e no MIT. Após um contato, a Xerox se negou a fornecer a Stallman e ao MIT o código do driver. Em razão disso, o problema ficou sem solução. A partir desse fato, Stallman concluiu que seria ingenuidade pensar que a assinatura de um termo de confidencialidade garantiria ajuda em qualquer circunstância, caso precisasse.

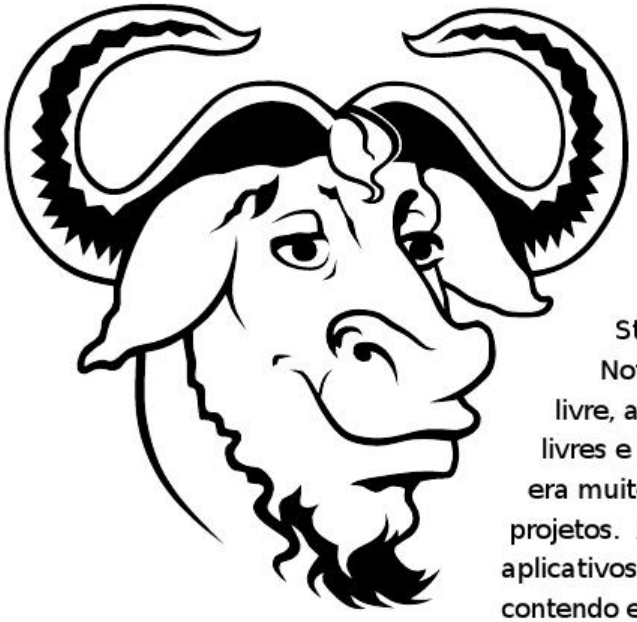
Uma ideia nunca abandonara a mente de Stallman: como fazer para que a comunidade de programadores voltasse a existir novamente? A resposta parecia óbvia. Tendo em vista que um computador só funciona se tiver um sistema operacional, era necessário um sistema operacional livre. O Unix já não o era mais. A única saída seria fazer um sistema operacional.

Depois de analisar a situação, Stallman concluiu que o novo sistema, para ser bom e versátil, deveria ser compatível com o Unix. Além disso, usuários do Unix gostariam de ter o seu próprio Unix, sem precisar comprá-lo. Isso aumentaria as chances de um trabalho em comunidade gerar algo satisfatório.

Em janeiro de 1984, Richard Stallman demitiu-se do MIT e começou a escrever o código do novo sistema. Nas suas palavras, “Foi necessário sair do MIT para ele não interferir na característica de Software Livre do novo projeto. Se eu tivesse permanecido no MIT, alguém poderia querer reivindicar algo sobre o projeto. Isso iria gerar um software proprietário, pois termos de uso acabariam sendo impostos. E o objetivo final, que era criar uma nova comunidade para a troca de software, não seria atingido.

Apesar de toda essa situação, Stallman foi convidado a continuar usando os recursos do Laboratório de Inteligência Artificial do MIT.

10-Projeto GNU



O sistema operacional de Richard Stallman recebeu o nome de Projeto GNU ou sistema operacional GNU. A palavra gnu, originalmente, refere-se a um mamífero ruminante, semelhante a um búfalo, com chifres espiralados, que vive no continente africano.

No caso do sistema operacional de Richard Stallman, GNU é um trocadilho que significa "Gnu's Not Unix" ou seja, o projeto GNU é uma concepção livre, ao contrário do Unix e de outros softwares, que eram livres e deixaram de sê-lo. Esse tipo de trocadilho, na época, era muito utilizado por programadores para nomearem seus projetos. Assim, o projeto GNU refere-se a uma série de aplicativos livres, desenvolvidos para os mais diversos fins, contendo editores de texto, planilhas de cálculo etc, tudo com o intuito de compor um sistema operacional livre.

Conta Stallman que o início do projeto foi um pouco conturbado. Ele ouvira falar de um tal Free University Compiler Kit, um compilador desenvolvido para múltiplas linguagens, incluindo C e Pascal. Richard escreveu para o autor perguntando se ele poderia inserir esse compilador no sistema operacional GNU. Obteve uma resposta debochada do autor, segundo o qual a universidade era free, mas o compilador não. Stallman, revoltado, iniciou o desenvolvimento do GNU pelo compilador. Assim, nasceu o compilador C chamado GCC (GNU C Compiler).

Entre o início e o fim do desenvolvimento do GCC, Stallman fez o GNU Emacs, um editor de textos muito utilizado até hoje. A sua elaboração começou em setembro de 1984. No início de 1985 o GNU Emacs já podia ser utilizado. O GNU Emacs já rodava com perfeição sobre o Unix e muitas pessoas pediram para usá-lo. Assim, Stallman o disponibilizou em um servidor ftp público do MIT, o prep.ai.mit. edu. Esse servidor está no ar até hoje e hospeda parte do projeto GNU. É possível logar-se a ele como usuário anonymous e navegar nos seus diretórios. Lá estará o projeto GNU!

11-Free Software Foundation

O interesse, por parte das pessoas, em usar o Emacs crescia cada vez mais. Alguns começaram a ajudar no seu desenvolvimento, o que era um objetivo antigo de Stallman.

Assim sendo, o inevitável ocorreu: era necessário injetar capital no projeto, que crescia cada vez mais e necessitava de colaboradores. A falta de capital para manter o projeto foi contornada com uma solução clássica: ainda em 1985, Richard Stallman fundou uma instituição chamada Free Software Foundation (FSF), criada para arrecadar fundos para a manutenção do Projeto GNU. A FSF existe até hoje e aceita doações. No entanto, a sua maior fonte de renda é originária da venda de CD-ROM com códigos-fonte e binários, além da venda de manuais impressos. Os empregados dessa instituição desenvolvem e mantêm vários programas e pacotes do sistema GNU, destacando-se as bibliotecas C e o Shell BASH, utilizado na maioria dos GNU/Linux.

É interessante dizer que todo sistema operacional possui um núcleo de controle, denominado kernel. O sistema operacional em si é constituído do kernel e de programas como editores de texto e utilitários de cópia de arquivos etc. O projeto GNU já possui vários programas, a maioria testados em Unix. No entanto, ainda não há um kernel maduro. Hurd é o nome do kernel que está em desenvolvimento e sem previsão para a primeira versão estável.

fisl14

14º Fórum Internacional

Software Livre

A tecnologia que liberta

Núcleo de Atendimento 14° Fórum Internacional Software Livre

Coordenação: Livia Araujo – livia@enfato.com.br – 51 8151 9026
Direção: Mariana Turkenicz– mariana@enfato.com.br – 51 8121.7062

Enfato multicomunicação

+55 (51) 30.261.261

Saiba mais:

fisl.org.br  @fisl_oficial  @fisl

Apoio Institucional

Promoção | Organização | Realização

Este material foi produzido e diagramado com softwares livres.