Por que você deve usar uma licença de estilo BSD em seu Projeto Open Source

Bruce Montague <brucem@alumni.cse.ucsc.edu >
Revisão: 52260

2018-09-15 17:33:40 por ebrandi.

Índice

1. Introdução

Este documento apresenta argumentos para a utilização de uma licença de estilo BSD para software e dados; especificamente recomenda utilizar uma licença de estilo BSD no lugar de uma GPL. Também pode ser lido como uma introdução e resumo das licenças de Projeto Open Source, BSD versus GPL.

2. Uma Breve História do Open Source

Muito antes do termo “Open Source” ser utilizado, o software era desenvolvido por associações livres de programadores e os softwares eram livremente trocados ou negociados. A partir do início dos anos 50, organizações como a SHARE e a DECUS desenvolviam grande parte do software que as empresas de hardware empacotavam em suas ofertas. Naquela época, as empresas de computadores estavam no negócio de hardware; qualquer coisa

que reduzisse o custo do software e disponibilizasse mais programas tornava as empresas de hardware mais competitivas.

Este modelo mudou nos anos 60. Em 1965, a ADR desenvolveu o primeiro produto de software licenciado e independente de uma empresa de hardware. A ADR estava competindo contra um pacote gratuito da IBM que foi originalmente desenvolvido por seus próprios clientes. A ADR patenteou seu software em 1968. Para interromper o compartilhamento de seu programa, eles forneceram seu software sob leasing de equipamento, na qual o pagamento foi distribuído durante a vida útil do produto. A ADR reteve a propriedade e pôde controlar a revenda e a reutilização.

Em 1969, o Departamento de Justiça dos EUA acusou a IBM de destruir negócios e empresas com seu agrupamento de software livre e hardware IBM. Como resultado deste processo, a IBM separou seu software; isto é, os softwares tornaram-se produtos independentes e separados do hardware.

Em 1968, a Informatics apresentou o primeiro software comercial revolucionário e rapidamente foi estabelecido o conceito do produto de software, da empresa de software e de taxas de retorno bem altas. A Informatics desenvolveu a licença perpétua que agora é comum em toda a indústria de computadores, onde a propriedade do software nunca é transferida para o cliente.

3. Unix de uma Perspectiva de Licenciamento BSD

A AT&T, que detinha a implementação original do Unix, era um monopólio regulado publicamente e amarrado a um tribunal anti-trust; ela foi legalmente impossibilitada de vender um produto no mercado de software. Entretanto, ela podia fornece-lo a instituições acadêmicas pelo preço da mídia.

As universidades adotaram rapidamente o Unix depois da divulgação de sua disponibilidade em uma conferência de Sistema Operacional. Foi extremamente útil o Unix rodar no PDP-11, um computador de 16 bits muito acessível na época, e que foi codificado em uma linguagem de alto nível, que era comprovadamente boa para a programação de sistemas. O DEC PDP-11 tinha, na verdade, uma interface de hardware aberta, projetada para tornar mais fácil para os clientes escreverem seus próprios sistemas operacionais, o que era comum. O famoso fundador da DEC, Ken Olsen, proclamou que “o software vem do céu quando você tem um bom hardware”.

O autor do Unix, Ken Thompson, retornou à Universidade da Califórnia de Berkeley (UCB) em 1975, e lecionou sobre o kernel linha por linha. Isso acabou resultando em um sistema evoluído conhecido como BSD (Berkeley Standard Distribution). A UCB converteu o Unix em 32 bits, adicionou memória virtual e implementou a stack TCP/IP, a qual foi essencialmente necessária para a construção da Internet. A UCB disponibilizou o BSD pelo custo da mídia, e isso ficou conhecido como “a licença BSD”. Um cliente comprava o Unix da AT&T e depois comprava uma fita BSD da UCB.

Em meados da década de 80, um processo anti-trust governamental contra a ATT resultou no seu desmembramento. A ATT ainda possuía o Unix e a partir daquele momento podia vendê-lo. Então a ATT embarcou em um esforço agressivo de licenciamento e a maioria dos Unixes comerciais da época tornaram-se derivações do Unix ATT.

No início dos anos 90, a ATT processou a UCB por violações de licenças relacionadas ao BSD. A UCB descobriu que a ATT havia incorporado, sem reconhecimento ou pagamento, muitas melhorias nos produtos da ATT originadas no BSD, e isso resultou em um longo processo judicial entre a ATT e a UCB. Durante esse período, alguns programadores da UCB embarcaram em um projeto para reescrever todos os códigos ATT que estavam associados ao BSD. Este projeto resultou em um sistema chamado BSD 4.4-lite (lite porque não era um sistema completo; faltavam 6 arquivos-chave ATT).

Uma longa série de artigos foram publicados um pouco depois disso na revista Dr. Dobbs, que descreviam uma versão do Unix derivada do BSD para PC 386, na qual os 6 arquivos que estavam faltando no 4.4 lite foram substituídos por arquivos de licença BSD. Este sistema, chamado 386BSD, foi devido ao ex programador da UCB, William Jolitz. Ele se tornou a base original de todos os BSD para PCs que estão atualmente em uso.

Em meados da década de 90, a Novell comprou os direitos do Unix da ATT e um acordo (então secreto) foi fechado para encerrar o processo. A UCB logo encerrou seu suporte para o BSD.

4. O Estado Atual das Licenças FreeBSD e BSD

A chamada nova licença BSD aplicada ao FreeBSD nos últimos anos é efetivamente uma afirmação de que você pode fazer qualquer coisa com o programa ou seu código fonte, mas você não tem nenhuma garantia e nenhum dos autores tem qualquer responsabilidade (basicamente, você não pode processar ninguém). Esta nova licença BSD destina-se a incentivar a comercialização de produtos. Qualquer código BSD pode ser vendido ou incluído em produtos proprietários, sem quaisquer restrições quanto à disponibilidade do seu código ou seu comportamento futuro.

Não confunda a nova licença BSD com "domínio público". Enquanto um item no domínio público também é gratuito para todos, ele não possui um proprietário.

5. As origens da GPL

Enquanto o futuro do Unix era tão confuso no final dos anos 80 e início dos anos 90, a GPL, um outro desenvolvimento com importantes considerações sobre licenciamento, surgiu.

Richard Stallman, o desenvolvedor do Emacs, era membro da equipe do MIT quando seu laboratório mudou de sistemas domésticos para sistemas proprietários. Stallman ficou chateado quando descobriu que não podia adicionar legalmente pequenas melhorias ao sistema. (Muitos dos colegas de trabalho de Stallman tinham saído para formar duas empresas com base em software desenvolvido no MIT e licenciado pelo MIT; parece ter havido desacordo sobre o acesso ao código-fonte desse software). Stallman criou uma alternativa para a licença de software comercial e a chamou de GPL, ou "GNU Public License". Ele também fundou uma fundação sem fins lucrativos, a Free Software Foundation (FSF), que pretendia desenvolver um sistema operacional completo, incluindo todos os softwares associados, e que não estaria sujeito a uma licença proprietária. Este sistema foi chamado de GNU, que significa "GNU is Not Unix".

A GPL foi projetada para ser o oposto da licença proprietária padrão. Para este fim, quaisquer modificações que eram feitas a um programa GPL tinham que ser devolvidas à comunidade GPL (exigindo que o código fonte do programa fosse disponibilizado para o usuário) e que qualquer programa que utilizar ou linkar com código GPL, teria que estar sob a GPL. A GPL pretendia impedir que o software se tornasse proprietário. Como o último parágrafo da GPL afirma:

"This General Public License does not permit incorporating your program into proprietary programs."[1]

A GPL é uma licença complexa, então aqui estão algumas regras básicas ao usar a GPL:

- você pode cobrar o quanto quiser para distribuir, dar suporte ou documentar o software, mas não pode vender o software em si.
- a regra básica indica que, se um código fonte GPL for necessário para um programa compilar, então o programa deve estar sob a GPL. Linkar estaticamente a uma biblioteca GPL requer que um programa esteja sob a GPL.
- a GPL exige que quaisquer patentes associadas ao software GPL sejam licenciadas para uso livre de todos.
- simplesmente agregar softwares juntos, como quando vários programas são colocados em um disco, não conta como inclusão de programas GPL em programas não-GPL.
- o que se resulta de um programa não conta como um trabalho derivado. Isso permite que o compilador gcc seja utilizado em ambientes comerciais sem problemas legais.
- como o kernel do Linux está sob a GPL, qualquer código estaticamente linkado ao kernel do Linux deve ser GPL. Este requisito pode ser contornado ao linkar dinamicamente módulos carregáveis do kernel. Isso permite que as empresas distribuam drivers binários, mas geralmente tem a desvantagem de que eles só funcionarão para versões específicas do kernel do Linux.

Devido em parte à sua complexidade, em muitas partes do mundo hoje as legalidades da GPL estão sendo ignoradas em relação ao Linux e softwares relacionados. As ramificações de longo prazo por causa disso não são claras.

6. As origens do Linux e da LGPL

Enquanto as guerras comerciais do Unix se intensificavam, o kernel do Linux foi desenvolvido como um clone do PC Unix. Linus Torvalds credita a existência do compilador GNU C e das ferramentas GNU associadas pela existência do Linux. Ele colocou o kernel do Linux sob a GPL.

Lembre-se de que a GPL requer que qualquer software que seja estaticamente linkado a um código GPL, também seja colocado sob a GPL. O código fonte desse software deve ser disponibilizado ao usuário do programa. O link dinâmico, no entanto, não é considerado uma violação da GPL. A pressão para colocar aplicativos proprietários no Linux tornou-se esmagadora. Tais aplicativos geralmente precisavam se linkar a bibliotecas do sistema. Isso resultou em uma versão modificada da GPL chamada LGPL ("Library", e depois renomeado para "Lesser", GPL). A LGPL permite que o código proprietário faça link com à biblioteca GNU C, glibc. Você não precisa liberar o código-fonte para um software que foi linkado dinamicamente a uma biblioteca LGPL.

Se você linkar estaticamente uma aplicação com a glibc, o que geralmente é necessário em sistemas embarcados, não será possível manter seu aplicativo proprietário, isto é, o código fonte deve ser liberado. Tanto a GPL quanto a LGPL requerem que qualquer software sob suas licenças liberem quaisquer modificações no código fonte.

7. Licenças Open Source e o Problema dos Softwares Orfãos

Um problema sério associado ao software proprietário é conhecido como “orphaning”. Isso ocorre quando um simples negócio falha ou quando uma mudança na estratégia de um produto faz com que uma cadeia de sistemas e empresas que dependiam deste produto, também falhem por motivos que estão fora de seus controles. Décadas de experiência mostraram que o tamanho ou o sucesso momentâneo de um fornecedor de software não é uma garantia de que seu software permanecerá disponível, pois as condições e estratégias atuais do mercado podem mudar rapidamente.

A GPL tenta impedir o software órfão cortando o link para a propriedade intelectual proprietária.

Uma licença BSD concede a uma pequena empresa o equivalente a um software-in-escrow sem quaisquer complicações ou custos legais. Se um programa licenciado pela BSD se torna órfão, uma empresa pode simplesmente assumir, de maneira proprietária, o programa do qual eles são dependentes. Uma situação ainda melhor ocorre quando uma base de código BSD é mantida por um pequeno consórcio informal, uma vez que o processo de desenvolvimento não depende da sobrevivência de uma única empresa ou de uma linha de produtos. A capacidade de sobrevivência da equipe de desenvolvimento quando eles estão mentalmente seguros é muito mais importante do que a simples disponibilidade física do código-fonte.

8. O que uma licença não pode fazer

Nenhuma licença pode garantir disponibilidade futura do software. Embora um detentor de direitos autorais possa tradicionalmente mudar os termos de um direito autoral a qualquer momento, a presunção na comunidade BSD é de que tal tentativa simplesmente faz com que o código fonte seja derivado (fork).

A GPL proíbe explicitamente a revogação da licença. Ocorreu no entanto, que uma empresa (Mattel) comprou um copyright GPL (cphack), e revogou todo o direito autoral, foi a tribunal e conseguiu prevalecer [2]. Ou seja, eles revogaram legalmente toda a distribuição e todos os trabalhos derivados com base nos direitos autorais. Se isso pode acontecer com uma distribuição maior e mais dispersa, fica uma questão em aberto; Há também alguma confusão sobre se o software estava realmente sob a GPL.

Em outro exemplo, a Red Hat comprou a Cygnus, uma empresa de engenharia que havia assumido o desenvolvimento das ferramentas de compilação da FSF. A Cygnus foi capaz de fazer isso porque eles desenvolveram um modelo de negócios no qual eles vendiam suporte para o software GNU. Isso permitiu que eles empregassem cerca de 50 engenheiros e os orientassem na direção dos programas, contribuindo com a preponderância de modificações. Como afirma Donald Rosenberg, "projetos usando licenças como a GPL ... vivem sob constante ameaça

de que alguém assuma o projeto produzindo uma versão melhor do código e fazendo isso mais rápido que os proprietários originais". [3]

9. Vantagens e Desvantagens da GPL

Um motivo comum para usar a GPL é ao modificar ou criar extensões ao compilador gcc. Isso é particularmente apropriado quando se trabalha com CPUs especiais únicas em ambientes em que todos os custos de software provavelmente são considerados como despesas gerais, com expectativas mínimas de que outros usarão o compilador resultante.

A GPL também é atraente para pequenas empresas que vendem CDs em um ambiente em que o "buy-low, sell-high" ainda pode dar ao usuário final um produto muito barato. Também é atraente para empresas que esperam sobreviver fornecendo várias formas de suporte técnico, incluindo documentação, para o mundo da propriedade intelectual GPL.

Um uso menos divulgado e não intencional da GPL é que ela é muito favorável a grandes empresas que querem minar empresas de software. Em outras palavras, a GPL é bem adequada para uso como arma de marketing, reduzindo potencialmente o benefício econômico geral e contribuindo para o comportamento monopolista.

A GPL pode representar um problema real para aqueles que desejam comercializar e lucrar com software. Por exemplo, a GPL aumenta a dificuldade que um estudante de pós-graduação terá em formar diretamente uma empresa para comercializar seus resultados de pesquisa, ou a dificuldade que um aluno terá em ingressar em uma empresa com a suposição de que um promissor projeto de pesquisa será comercializado.

Para aqueles que precisam trabalhar com implementações linkadas estaticamente em vários modelos de software, a GPL é geralmente uma licença ruim, porque impede o uso de implementações proprietárias dos modelos. A GPL minimiza, assim, o número de programas que podem ser compilados usando o modelo GPL. A GPL tinha como objetivo não fornecer um mecanismo para desenvolver um padrão na engenharia de produtos proprietários. (Isso não se aplica a aplicativos Linux porque eles não usam links estáticos, em vez disso, usam uma trap-based API.)

A GPL tenta fazer com que os programadores contribuam para um conjunto de programas em desenvolvimento, para então competir na distribuição e suporte deste conjunto. Essa situação não é realista para muitos dos padrões de sistema exigidos, que podem ser aplicados em ambientes amplamente diferentes, e que exigem personalização comercial ou integração com padrões legados sob licenças existentes (não-GPL). Os sistemas real-time usam frequentemente links estáticos, de modo que a GPL e a LGPL são definitivamente consideradas problemas potenciais por muitas empresas de sistemas embarcados.

A GPL é uma tentativa de manter os trabalhos disponíveis, independentemente da demanda nos estágios de pesquisa e desenvolvimento. Isso maximiza os benefícios para pesquisadores e desenvolvedores, a um custo desconhecido para aqueles que se beneficiariam de uma distribuição mais ampla.

A GPL foi projetada para impedir que os resultados de uma pesquisa sejam transferidos para produtos proprietários. Este passo é frequentemente considerado o último passo no pipeline tradicional de transferência de tecnologia e é geralmente o mais difícil mesmo sob as melhores circunstâncias; a GPL pretendia tornar isso impossível.

10. Vantagens da licença BSD

Uma licença de estilo BSD é uma boa opção para pesquisas de longa duração ou outros projetos que precisam de um ambiente de desenvolvimento que:

- tem custo próximo a zero
- irá evoluir durante um longo período de tempo
- permite que qualquer pessoa mantenha a opção de comercializar os resultados finais com problemas legais mínimos.

Esta consideração final pode muitas vezes ser a dominante, como foi quando o projeto Apache decidiu sua licença:

“Este tipo de licença é ideal para promover o uso de um corpo de referência de código que implementa um protocolo para um serviço comum. Esta é outra razão pela qual a escolhemos para o grupo Apache - muitos de nós queriam que o HTTP sobrevivesse e se tornasse um verdadeiro padrão multipartidário, e não nos importaríamos nem um pouco se a Microsoft ou a Netscape escolhessem incorporar nosso mecanismo HTTP ou qualquer outro componente de nosso código em seus produtos, se isso ajudasse a manter o objetivo comum de manter o HTTP universal... Tudo isso significa que, estrategicamente falando, o projeto precisa manter ímpeto suficiente e que os participantes percebam um maior valor contribuindo com seu código para o projeto, mesmo código que teria valor se fosse mantido proprietário.”

Os desenvolvedores tendem a achar a licença BSD atrativa, pois ela mantém os problemas legais fora do caminho e permite que eles façam o que quiserem com o código. Em contraste, aqueles que esperam principalmente usar um sistema em vez de programá-lo, ou que esperam que outros evoluam o código, ou aqueles que não esperam ganhar a vida com seu trabalho associado ao sistema (como funcionários do governo), achem a GPL atraente, porque força o código desenvolvido por outros a ser dado a eles de volta e impede que os seus empregadores retenham os direitos autorais e, portanto, potencialmente "enterra" o problema de software órfão. Se você quiser forçar seus concorrentes a ajudá-lo, a GPL é atraente.

Uma licença BSD não é simplesmente um presente. A pergunta “por que devemos ajudar nossos concorrentes ou deixá-los roubar nosso trabalho?” surge frequentemente em relação a uma licença BSD. Sob uma licença BSD, se uma empresa vier a dominar um nicho de produto que outros consideram estratégico, as outras empresas podem, com esforço mínimo, formar um mini consórcio visando restabelecer a paridade, contribuindo para uma variante BSD competitiva que aumente a competição e a justiça no mercado. Isso permite que cada empresa acredite que será capaz de lucrar com alguma vantagem que ela possa proporcionar, ao mesmo tempo em que contribui para a flexibilidade e eficiência econômica. Quanto mais rápido e fácil os membros cooperantes puderem fazer isso, maior sucesso eles terão. Uma licença BSD é essencialmente uma licença minimamente complicada que permite tal comportamento.

Um efeito chave da GPL é fazer com que um sistema Open Source completo e competitivo seja amplamente disponibilizado ao custo de mídia, e isso é uma meta razoável. Uma licença no estilo BSD, em conjunto com consórcios ad-hoc de indivíduos, pode atingir essa meta sem destruir as premissas econômicas construídas em torno da implementação final do pipeline de transferência de tecnologia.

11. Recomendações Específicas para usar uma licença BSD

- A licença BSD é preferível para a transferência de resultados de pesquisa de uma maneira que seja largamente implantada e que mais beneficie uma economia. Como tal, as agências de financiamento de pesquisa, como a NSF, ONR e DARPA, devem encorajar nas fases iniciais dos projetos de pesquisa financiados, a adoção de licenças de estilo BSD para software, dados, resultados e hardware aberto. Eles também devem incentivar a formação de padrões baseados em sistemas Open Source implementados e projetos Open Source em andamento.

- A política do governo deve minimizar os custos e as dificuldades de passar da pesquisa para a implantação. Quando possível, os subsídios devem exigir que os resultados estejam disponíveis sob uma licença de estilo BSD amigável à comercialização.

- Em muitos casos, os resultados de longo prazo de uma licença de estilo BSD refletem com mais precisão os objetivos proclamados na carta de pesquisa das universidades do que ocorre quando os resultados são protegidos por direitos autorais ou patenteados e sujeitos ao licenciamento universitário proprietário. Existem evidências casuais de que as universidades são financeiramente mais bem recompensadas a longo prazo, divulgando resultados de pesquisa e apelando para doações de ex-alunos de sucesso comercial.

- As empresas há muito reconheceram que a criação de padrões de facto é uma técnica de marketing fundamental. A licença BSD serve bem a essa função se uma empresa tiver realmente uma vantagem exclusiva na evolução do sistema. A licença é legalmente atraente para o público mais amplo, enquanto a expertise da empresa garante o seu controle. Há momentos em que a GPL pode ser o veículo apropriado para uma tentativa de criar tal

padrão, especialmente quando se tenta prejudicar ou cooptar outras pessoas. A GPL, no entanto, penaliza a evolução desse padrão, porque promove um conjunto em vez de um padrão comercialmente aplicável. O uso de tal conjunto constantemente sofre um aumento de problemas legais e comerciais. E pode não ser possível misturar padrões quando alguns estão sob a GPL e outros não. Um verdadeiro padrão técnico não deve obrigar a exclusão de outros padrões por razões não técnicas.

- As empresas interessadas em promover um padrão em evolução, que pode se tornar o núcleo dos produtos comerciais de outras empresas, devem ter cuidado com a GPL. Independentemente da licença usada, o software resultante geralmente será transferido para quem realmente faz a maioria das alterações de engenharia e que mais entende o estado do sistema. A GPL simplesmente adiciona mais atrito legal ao resultado.

- Grandes empresas, nas quais código Open Source é desenvolvido, devem estar cientes de que os programadores apreciam o Open Source porque ele deixa o software disponível para o funcionário quando ele mudar de empregador. Algumas empresas encorajam esse comportamento como uma vantagem de emprego, especialmente quando o software em questão não é diretamente estratégico. Trata-se, na verdade, de um benefício antecipado com possíveis custos de oportunidade perdidas, mas sem custos diretos. Incentivar os funcionários a trabalhar pela aclamação dos colegas fora da empresa é um benefício barato que uma uma empresa pode, por vezes, fornecer com desvantagem quase zero.

- Pequenas empresas com projetos de software vulneráveis ao software órfão, devem tentar usar a licença BSD sempre que possível. Empresas de todos os portes devem considerar a formação de tais projetos Open Source quando for vantajoso manter mínimas as despesas legais e organizacionais associadas a um verdadeiro projeto Open Source de estilo BSD.

- As organizações sem fins lucrativos devem participar de projetos Open Source sempre que possível. Para minimizar os problemas de engenharia de software, como a mistura de código sob diferentes licenças, as licenças no estilo BSD devem ser incentivadas. Desconfiar da GPL deve ser particularmente o caso de organizações sem fins lucrativos que interagem com o mundo de desenvolvimento. Em alguns locais onde a aplicação da lei se torna um exercício caro, a simplicidade da nova licença BSD, em comparação com a GPL, pode ser de considerável vantagem.

12. Conclusão

Em contraste com a GPL, que é projetada para impedir a comercialização proprietária do código Open Source, a licença BSD impõe restrições mínimas sobre o comportamento futuro. Isso permite que o código BSD permaneça como código aberto ou se integre a soluções comerciais, à medida que as necessidades de um projeto ou empresa mudam. Em outras palavras, a licença BSD não se torna uma bomba-relógio legal em nenhum ponto do processo de desenvolvimento.

Além disso, como a licença BSD não vem com a complexidade legal das licenças GPL ou LGPL, ela permite que desenvolvedores e empresas gastem seu tempo criando e promovendo um bom código, em vez de se preocupar se esse código viola algum licenciamento.

13. Adendos

```
[1] http://www.gnu.org/licenses/gpl.html

[2] http://archives.cnn.com/2000/TECH/computing/03/28/cyberpatrol.mirrors/

[3] Open Source: the Unauthorized White Papers, Donald K. Rosenberg, IDG Books,
    2000. Quotes are from page 114, ``Effects of the GNU GPL''.

[4] In the "What License to Use?" section of
    http://www.oreilly.com/catalog/opensources/book/brian.html

This whitepaper is a condensation of an original work available at
```

```
http://alumni.cse.ucsc.edu/~brucem/open_source_license.htm
```