

Licenças de software livre

descrição, sistematização, compatibilidade e incompatibilidades

*Prof. Dr. Fabio Kon, Nelson Lago
e Vanessa Sabino*

*Centro de Competência em Software Livre
IME-USP*

29/08/2011 - SERPRO



- **direitos autorais e software**
- **origem das licenças de software livre**
- **licenças e contratos**
- **classificação das licenças de software livre**
- **descrição de algumas licenças**
- **(in)compatibilidades**
- **conclusão**

- **direitos autorais e software**
- **origem das licenças de software livre**
- **licenças e contratos**
- **classificação das licenças de software livre**
- **descrição de algumas licenças**
- **(in)compatibilidades**
- **conclusão**

- **Propriedade física X “propriedade intelectual”**
 - objetos \neq ideias
- **Desenvolvimento do conhecimento depende do compartilhamento de ideias**
- **Desenvolver novas ideias depende também de dinheiro**
- **Solução de compromisso: *copyright*, patentes**
 - um mecanismo para viabilizar a criação de novas ideias e ao mesmo tempo promover o compartilhamento dessas ideias

- **Lei americana:** a constituição explicitamente afirma que as proteções garantidas pelo copyright e pelo sistema de patentes têm tempo limitado e existem para promover o progresso
- **Leis brasileira e europeia:** no caso do copyright, o indivíduo tem maior controle sobre sua criação
 - Está em análise uma reforma na lei de copyright brasileira que torna mais explícita a função social do copyright de maneira similar à lei americana
- **Nos dois casos, trata-se de direitos outorgados pelo estado e não de “direitos naturais”;** tanto que passam para domínio público e têm restrições através do “fair use”, diferentemente da propriedade física

- **Software é um caso especial:**
 - É um meio de produção, diferentemente de um livro ou uma patente;
 - Ainda assim, é abstrato;
 - Poucas “invenções” são de fato patenteáveis; além disso, a complexidade reduz grandemente a relevância de invenções isoladas
 - Patentes não se aplicam diretamente ao software porque normalmente o fonte é secreto
 - A ideia de “propriedade intelectual” é uma analogia; houve (e há) dúvidas quanto à melhor analogia a ser usada para o software (copyrights, patentes, nada...)

- **Ainda assim, o software é tipicamente protegido por copyright (por 95 anos) e por patentes nos EUA (por 17 anos)**
- **Além disso, o código fonte é secreto**
- **O resultado é que há um desequilíbrio entre o interesse público e o privado**

- **O sistema de copyright em si tem problemas, pois foi criado para outra época:**
 - O “custo social” do copyright hoje é muito maior que no século XIX, pois antigamente apenas os editores eram afetados
 - Apesar da velocidade dos avanços tecnológicos, a duração do copyright tem crescido
 - “Lobbies” procuram confundir copyright com propriedade, através do uso de expressões como “pirataria”, “propriedade intelectual” etc.

- **A solução de compromisso do *copyright* procurava equilibrar o interesse privado e o interesse público**
- **Hoje em dia, mudanças na legislação e na tecnologia destruíram esse equilíbrio**
- **Reação: surgimento da comunidade de software livre**
 - retorno ao compartilhamento (do código-fonte) e à colaboração (troca de ideias e trabalho em equipe)
 - só é possível em um ambiente que facilite a troca de código-fonte; por isso, o crescimento junto com a Internet

- direitos autorais e software
- **origem das licenças de software livre**
- licenças e contratos
- classificação das licenças de software livre
- descrição de algumas licenças
- (in)compatibilidades
- conclusão

- **Mas o contexto é outro**
 - software não-livre é a norma
 - há leis rígidas sobre *copyright*
 - patentes de software (inválidas no Brasil, porém...) afetam o desenvolvimento
 - pessoas diferentes têm visões diferentes
- **O que é o “certo” nesta realidade?**
 - todo software deveria ser livre?
 - “liberdade” significa que tudo pode ser feito, inclusive “fechar” o software?
 - misturas de software livre e não-livre são boas?

- **DEPENDENTE!**

- diferentes pessoas têm diferentes opiniões
- diferentes programas ou situações podem se beneficiar de diferentes abordagens

- **Quem decide?**

- dentro do nosso contexto social e cultural, só há uma resposta razoável: o autor
- a escolha por parte do autor também é adequada do ponto de vista legal, já que ele é o detentor do *copyright*

- **Graças à lei, todo trabalho de criação (incluindo software) é automaticamente sujeito ao *copyright***
 - para que alguém diferente do autor possa fazer praticamente qualquer tipo de uso desse trabalho, é preciso haver uma permissão explícita por parte do autor
 - tipicamente, essa permissão é formalizada através de uma licença ou contrato escrito
 - essa licença ou contrato é um documento jurídico e, como tal, precisa ser razoavelmente detalhada e precisa
 - o autor é quem define os termos dessa licença ou contrato

- **No entanto, em geral os desenvolvedores de software livre evitam redigir “do zero” as licenças detalhando essas condições**
 - dá trabalho escrever uma licença consistente do ponto de vista jurídico
 - não é necessário “reinventar a roda”
 - se o objetivo é o compartilhamento do código, códigos diferentes com condições de uso e distribuição diferentes atrapalham
- **Por isso, existem licenças comumente usadas por programadores com visões e interesses similares dependendo do contexto e do projeto**
- **Ainda assim, existem mais de 70 licenças “comumente usadas” registradas no sourceforge!**

- direitos autorais e software
- origem das licenças de software livre
- **licenças e contratos**
- classificação das licenças de software livre
- descrição de algumas licenças
- (in)compatibilidades
- conclusão

- **Um “contrato” nos EUA corresponde ao que no Brasil é chamado “contrato formal”**
- **Um contrato desse tipo envolve:**
 - contrapartidas de ambas as partes
 - formalização do aceite por ambas as partes
- **No caso do software, um contrato típico é aquele através do qual o direito de uso de um software (que é proibido sem permissão do autor) é oferecido como contrapartida a pagamento**

- **No caso de mera cessão unilateral de direitos, a lei americana considera tratar-se de uma *licença*; a lei Brasileira, “contrato benéfico”**
 - Não há necessidade de formalização por escrito
 - Legislações referentes a proteção do consumidor têm muito menos impacto
 - Muitas “licenças” de software restrito são, na verdade, inválidas, pois deveriam ser contratos formais
- **Os limites entre os dois às vezes são pouco claros**

- **As licenças de software livre são facilmente enquadradas como licenças ou contratos benéficos**
 - Eventuais condições de licenciamento, como as contidas na GPL, não são contrapartidas no sentido jurídico
 - A legislação mais relevante não é a referente a contratos, mas sim a copyrights (pois a licença é uma cessão de direitos garantidos pelo copyright)
 - Portanto, infringir uma licença de software livre em geral equivale a infringir a legislação de copyright, que é bem conhecida e razoavelmente homogênea em quase todas as jurisdições (Convenção de Berna)
 - Essa é uma das vantagens dessa caracterização

- direitos autorais e software
- origem das licenças de software livre
- licenças e contratos
- **classificação das licenças de software livre**
- descrição de algumas licenças
- (in)compatibilidades
- conclusão

- **Apesar do grande número de licenças diferentes, pode-se agrupá-las em 3 categorias principais:**
 - recíprocas totais (GPL, AGPL e similares): o software é livre, deve permanecer livre e trabalhos derivados devem ser também livres
 - recíprocas parciais (LGPL, EPL, MPL e assemelhadas): o software é livre e deve permanecer livre, mas trabalhos que o usam não precisam ser livres
 - permissivas (Apache, MIT/X11, BSD e assemelhadas): o software é livre, mas pode ser relicenciado sem permissão adicional do autor

- **Exigem que trabalhos relacionados ao trabalho original de alguma maneira sejam licenciados sob a mesma licença do trabalho original**
- em alguns casos pode haver dúvidas sobre o “relacionamento” entre os trabalhos
 - a GPL explicitamente menciona o processo de “linkagem”, típico de programas que rodam em uma única máquina escritos em linguagens como C/C++
 - no contexto de aplicativos CORBA ou com Web Services, por exemplo, a GPL pode ser interpretada de diferentes maneiras
 - tradicionalmente, entende-se que interfaces desse tipo tornam os trabalhos independentes; mas pode-se tentar usar isso como mecanismo para burlar a licença
 - no entanto, no caso de uma disputa legal, o espírito da licença (por exemplo, o preâmbulo da GPL) pode ser usado em juízo para resolver a questão

- **Exigem que o código licenciado em si e modificações e melhorias diretamente relacionadas a ele sejam sempre disponíveis sob a mesma licença**
- **Permitem que projetos independentes que apenas utilizem as funcionalidades desse código sejam licenciados de forma independente**
- **Usadas tipicamente para bibliotecas, frameworks e ambientes como JBoss**
- **Pode-se pensar em “licença baseada em arquivos”:** os arquivos que correspondem ao código-fonte devem ser sempre livres
- **Pode haver ainda mais casos dúbios e maneiras de burlar a licença**

- **Permitem o relicenciamento sem permissão adicional do autor com mínimas restrições**
- **Interessantes quando o objetivo é disseminar o software e a tecnologia correspondente, independente da licença final (livre ou não)**
 - exemplos: a implementação TCP/IP do BSD, que foi a primeira utilizada no linux mas também foi a utilizada no windows 95/98, NT e XP; ogg vorbis; BIND
- **Pode haver um equilíbrio entre versões abertas e fechadas do mesmo código através da definição de marcas fortes, como ocorre com o apache**
 - a versão aberta do apache é amplamente conhecida, tornando pouco interessante a criação de derivados fechados

- direitos autorais e software
- origem das licenças de software livre
- licenças e contratos
- classificação das licenças de software livre
- **descrição de algumas licenças**
- (in)compatibilidades
- conclusão

- **BSD: Muito usada, porque era a licença do BSD Unix e é bastante simples**
- **Originalmente continha a “cláusula de propaganda”, que causou mais problemas que vantagens**
- **Atualmente, há duas variantes sem essa cláusula normalmente utilizadas (“Nova BSD” ou “BSD simplificada”)**
- **As duas são iguais, mas uma delas tem uma cláusula adicional que proíbe o uso do nome do autor para promover produtos derivados**

A licença Apache 2.0

- **Bastante detalhada do ponto de vista jurídico**
 - Elenca e explicita todos os direitos transferidos pela licença
- **Fortemente centrada na legislação americana**
- **Procura proteger a marca “apache” mas sem impor restrições exageradas para seu uso**
- **Incompatível com a GPL versão 2 (mas compatível com a versão 3)**
- **Licenciamento automático de patentes**

- **60% do código de uma distribuição típica é GPL**
 - a segunda licença mais popular é a LGPL, com 7%
- **GPL só versa sobre a distribuição; qualquer uso é permitido, incluindo combinações com softwares restritos**
 - não há razão para disputas legais referentes ao uso
- **a única maneira de distribuir código GPL de maneira legal é atendendo à GPL**
 - Violar a GPL é violar a lei de copyright, que é bem estabelecida, conhecida e aceita
 - As condições de redistribuição não são contrapartidas, pois o “beneficiário” geralmente não é o autor original

- **Dúvidas sobre a “validade legal” da GPL são basicamente propaganda enganosa**
- **O tamanho do código original em relação ao código que o utiliza é irrelevante**
 - o código GPL pode consistir de 100 linhas e o programa que o utiliza pode consistir de 2M de linhas; as regras da licença são as mesmas
- **O tipo de relacionamento entre os pedaços de código é irrelevante**
 - não importa se o código GPL é responsável apenas por uma funcionalidade periférica do programa total; as regras da licença são as mesmas
- **A regra vale mesmo que do ponto de vista legal não se trate de um trabalho derivado!**
- **Exemplo: python e readline**

- **“Viralidade” da GPL**

- “Não é justo que meu software de 2M de linhas de código deva ser obrigatoriamente GPL por causa do uso de 100 linhas de código GPL”

- **No entanto**

- ninguém é obrigado a usar código GPL e, portanto, a licenciar nada sob a GPL; mesmo “erros” são tolerados se corrigidos quando identificados
- se um código GPL não tem valor suficiente que justifique o licenciamento de um produto sob a GPL, basta não utilizar esse código e reescrevê-lo ou obter código similar de outro fornecedor
- licenças de software restritivas (pagas ou não) também impõem regras às vezes “desagradáveis”

- **“...ou qualquer versão posterior, conforme você preferir...”**
- **Discussão com a comunidade durante 18 meses**
- **Permite combinações com licenças em que os termos são similares mas diferentes para remover incompatibilidades desnecessárias (em particular, Apache 2.0)**
 - Variações na exclusão de garantias
 - Variações nas formas de atribuição
 - Restrições ao uso de marcas registradas
 - etc
- **Proíbe “tivoization” em “produtos de usuário”**
- **Cláusula de retaliação por patentes**
- **DMCA, DRM e “sistemas efetivos de proteção”**

- **Variação da GPL que aborda programas acessíveis via rede**
- **se o código original inclui um mecanismo para que seu fonte possa ser baixado automaticamente pelo usuário, variações do código necessariamente precisam incluir um mecanismo similar**
- **o lauchpad foi disponibilizado sob essa licença**
- **com o crescimento de aplicações de internet “ricas” e da “web 2.0”, é possível que essa licença venha a se popularizar**

- **Variação da GPL voltada para bibliotecas, compatível com a GPL**
- **Permite a utilização com programas não-livres, mas exige que a biblioteca em si permaneça livre**
- **Procura explicitar várias formas de relação permitidas e esclarecer detalhes como o copyright que se aplica aos arquivos de cabeçalho (“includes”) etc.**
- **O texto é muito focado na ideia de biblioteca e o uso em outros contextos (como frameworks) é um pouco “forçado”**

- **Nasceu com a transformação do Netscape em software livre**
- **Muito usada e a base para muitas licenças similares**
- **“Baseada em arquivos”**
- **Incompatível com a GPL e com a Apache 2.0 (!!!)**
- **Por outro lado, mais simples e mais abrangente que a LGPL**
 - O texto não se prende à ideia de biblioteca X código que utiliza a biblioteca
- **Versão 2.0 a caminho, provavelmente resolverá os problemas de compatibilidade**
 - Para Simon Phipps, da OSI, provavelmente a melhor escolha entre as recíprocas parciais

- direitos autorais e software
- origem das licenças de software livre
- licenças e contratos
- classificação das licenças de software livre
- descrição de algumas licenças
- **(in)compatibilidades**
- conclusão

- **Diferentes condições podem tornar a mistura de códigos com licenças diferentes ilegal**
 - a “proliferação de licenças” é ruim para a comunidade
- **Problemas de compatibilidade são comuns com a GPL, por causa do mecanismo de “copyleft”**
 - muitas vezes, detalhes legais, como cláusulas que definem um foro específico para resolução de conflitos
- **OSI classifica várias licenças explicitamente como “redundantes”**
 - muitas são equivalentes em intenção, mas ainda assim incompatíveis
- **Solaris e Linux não podem usar código um do outro por incompatibilidade entre as licenças**

- **Problemas podem ser sutis**
 - o uso da Qt (não-livre na época) pelo KDE
 - teoricamente, KDE só pode ser distribuído sob a GPL3
 - openssh: código livre é livre para sempre
 - EPL é incompatível com LGPL 2/3 e GPL 2/3 (!!!)
 - mplayer e sistemas embarcados
 - roteadores D-Link e GigaByte

- direitos autorais e software
- origem das licenças de software livre
- licenças e contratos
- classificação das licenças de software livre
- descrição de algumas licenças
- (in)compatibilidades
- **conclusão**

- **Identificar compatibilidade ou não entre licenças é complexo e pode haver impacto jurídico**
- **A escolha de uma licença para um novo projeto deve ser feita com cuidado e critério**
 - mudanças posteriores podem ser difíceis se houver muitos contribuidores (ex: linux)
 - vale muito a pena ser compatível com a GPL
 - facilidade para agregar código alheio
 - dificilmente há boa razão *prática* para não ser compatível
 - compatibilidade pode ser de “mão-única” (como no caso do FreeBSD X Linux)
- **No entanto, é importante lembrar que isso é praticamente irrelevante para o usuário final: o mero uso dificilmente gera situações ilegais**