UMA DEFINIÇÃO INFORMAL DE DOCUMENTAÇÃO: ANÁLISE SEMIÓTICA

Ana Cristina Fricke Matte – UFMG

RESUMO: A documentação, para o software livre, é absolutamente essencial. A partir de uma analogia entre documentação de software e documentação para alugar uma casa, o texto analisa semioticamente a documentação como elemento de comunicação, discutindo o papel dos tipos de documento nos processos de produção e divulgação de software.

PALAVRAS-CHAVE: Documentação. Software livre. Semiótica. Comunicação.

ABSTRACT: The free software documentation is absolutely essential. From an analogy between software's documents and the necessary documents to rent a house, the text analyses in a semiotic field the documentation as part of a communication process, discussing the role of the kind of documents in the production and promulgation of free software.

KEYWORDS: Documentation. Free software. Semiotics. Communication.

Introdução

Falar em documentação no campo do software livre é usual e rotineiro, no entanto torna-se um assunto que beira o obscuro quando se trata de prioridades na implementação de software. Existem comunidades que levam a documentação muito a sério, outras documentam porque acham obrigatório, mas se pudessem não fariam, e ainda há outras que evitam documentar, propositadamente ou não (e isso de fato não importa).

RTFM (read the fucking manual) é uma expressão muito comum nesse meio, assim como "consulte o amigo google" ou "google it". Ambas pressupõem que haja informações suficientes e disponíveis tanto para usuários leigos quanto para usuários avançados. Num ambiente online como os chats e fóruns de suporte de software livre, nos quais o usuário busca ajuda, parece estranho e mesmo indelicado que as pessoas respondam com um RTFM: por que existem esses ambientes de ajuda se não ajudam? Mas mesmo essa expressão, no contexto da definição de documentação para o software livre, possui uma base lógica que a torna, em alguns casos, absolutamente plausível.

Levemos em consideração que há dois princípios básicos altamente vitais para o software livre: produção colaborativa e divulgação ampla. Sempre que existe mais de uma pessoa desenvolvendo um trabalho (software ou qualquer outro), a comunicação é fundamental. Existem diversas maneiras de trabalhar em conjunto, mas quando o objeto final não pode ser segmentado

para que cada um realize uma tarefa individual sem que ela afete o conjunto (o que talvez seja mesmo impossível), registrar e disponibilizar mudanças, coisas por fazer, decisões, discussões e o próprio objeto do trabalho é crucial. Em outras palavras, documentar o processo e o final do processo é fundamental, assim como divulgar os resultados é vital para a existência de um objeto livre, seja ele um software, um artigo ou uma obra de arte.

Sendo assim, deveríamos pressupor que a própria existência de uma cultura livre impõe a pré-existência de documentação correlata. É esse jogo de pressuposições incrustado na comunicação que vamos estudar com mais atenção neste trabalho.

RTFM vs. Google It!

Vamos começar analisando a esfera dessas duas expressões bastante conhecidas de quem já procurou ajuda em fóruns e chats de suporte de software livre.

O RTFM (Read The fucking Manual: http://pt.wikipedia.org/wiki/Rtfm) é, muitas vezes, tão delicado quanto "ora, vá catar coquinho".

A tradução para a Wikipédia brasileira do artigo correspondente, encontrado na Wikipédia inglesa, discute justamente essas possibilidades de leitura.

O princípio fundamental do software livre, não canso de dizer, é compartilhar conhecimento. Qual não será a reação de alguém que, como único local de suporte para o software livre que está aprendendo a usar, conhece um fórum ou uma sala de IRC e recebe um RTFM como resposta à sua pergunta? O que o sujeito "informante" espera? O que o sujeito "perguntante" entende?

Qualquer comunicação é afetada por diferentes tipos de ruídos (ver esquema de comunicação de Silva em BARROS, 2002). Um ruído importante é justamente a diferença entre aquilo que se pretende comunicar e aquilo que é efetivamente comunicado. Essa lacuna faz parte do processo comunicativo, pois baseia-se nos pressupostos que seriam ou não necessário ressaltar. Em outras palavras: eu não preciso dizer que o personagem levantou o pé, baixou no primeiro degrau, levantou o outro, baixou no degrau seguinte e assim por diante até chegar ao topo da escada para que alguém entenda que isso foi feito quando eu digo "subiu a escada". Neste exemplo mesmo, se eu dissesse "subiu a escada voando", dispensaria essa mesma informação extra pressupondo que o destinatário da mensagem soubesse que "voando" significa "rapidamente" e não "voando como um jato ou uma ave". Mas isso não impediria alguém de supor que "voando" tivesse o segundo sentido, por exemplo se eu estivesse contando uma história em que houvesse outros elementos mágicos ou

paranormais.

A comunicação, portanto, baseia-se num jogo de pressuposições. O destinador “informante” pode dizer “RTFM” sem a menor intenção de ofender, simplesmente pressupondo que o destinatário “perguntante” saiba que a independência gerada pelo costume de procurar respostas nos documentos existentes é de alto valor nesse campo – em outras palavras: no software livre, o usuário é estimulado a obter conhecimento para resolver e buscar soluções sozinho, de modo a contribuir com sua própria criatividade, além de “amadurecer” sua condição, passando de usuário leigo para experimentado.

Mas isso não significa que, ao ler um “RTFM”, eu possa ter certeza de que a intenção era essa. Afinal, pode ser que fosse, de fato, um “ora vá catar coquinho” disfarçado (ou nem tão disfarçado). Falhas de comunicações decorrentes dessas possíveis leituras são absolutamente normais em qualquer comunicação, por isso geralmente as pessoas interessadas em que o conteúdo de sua mensagem seja maximamente compreendido duplicam, aumentam, exageram as pistas sobre o exato conteúdo a ser comunicado. Mesmo assim, não há garantias. A língua é um produto “vendido” sem qualquer garantia.

Diante de um “RTFM”, o sujeito “perguntante” pode reagir de diversas formas, todas plausíveis. Há os que desistem. Outros, com a maior paciência, obedecem: lêem o manual, geralmente em inglês e muito técnico para um iniciante, ou vão procurar no google, sem saber bem o que procurar no meio do meio milhão de respostas que o sábio google retorna. Se este for o caso, voltam pro fórum ou chat e dizem: “fiz, não adiantou”.

O que vem depois pode variar entre um destinador “informante” que começa a “tirar sarro” do “pobre coitado que está lá a mostrar sua santa incompetência, batman”, até, mais raramente, alguém que diz: "porque não procura assim..." ou "tem esse outro manual" ou "tente então explicar melhor seu problema".

O artigo da Wikipedia, que fiz questão de traduzir para o português (e que Eduardo Henrique Rivelli Pazos gentilmente redirecionou de Rtfm para RTFM) também afirma que entre hackers essa expressão pode ser usada como um aviso a longo prazo, já que os hackers devem ser caracterizados por sua independência e iniciativa. Essa idéia de que o usuário deve aprender a postura “hacker” é também muito comum nesse meio.

A leitura do manual não deve ser dispensada a priori, ela pode ser útil e mais será quanto mais pessoas ajudarem a pensá-lo e repensá-lo. É extremamente saudável estimular as pessoas a buscarem soluções em muitos meios diferentes, não só perguntando a outros usuários, nem sempre

imediatamente disponíveis. E, evidentemente, isso não se aplica somente a hackers.

Mas nesse ponto entra uma outra questão relevante. A história do software livre criou um tipo de status altamente positivo para os hackers (hackers – os "fuçadores" –, não crackers – violadores de leis –), status esse desejado por muitas das pessoas que entram no meio. Quando o destinador "informante" usa um "RTFM" pode perfeitamente estar usando do pressuposto "eu já li, eu já conheço, não tenho tempo para responder algo tão óbvio" para mostrar uma superioridade em relação ao destinatário "perguntador".

Essa posição vai contra a própria história do software livre, que hoje busca ampliar o número de usuários atingindo cada vez mais pessoas que não sejam necessariamente especialistas em computação. Portanto, manter esse status às custas da expulsão das pessoas "normais" dos meios de contato e suporte (fóruns e chats) é mais que um contra senso: é uma subversão dos valores do software livre. É repetir a revolução industrial, aquela que tirou o poder dos feudos para colocar nas mãos da classe média, mantendo acorrentados e empobrecidos aqueles trabalhadores que, com sua força, garantiram a vitória da revolução.

As comunidades de software livre são compostas por pessoas de todos os tipos, no que diz respeito a essa utilização do "RTFM" (lembrando que sempre há os que evitam o uso da expressão para evitar sua possível conotação negativa). O "google it!" ou "vá procurar no santo Google" (santo, amigo ou simplesmente google) é menos carregado, mas não muito menos. O Google pode permitir acesso a milhares de documentos sobre o assunto problemático. Mas usar o Google também requer conhecimento de causa. Conhecimento técnico do próprio sistema de buscas, que pode ajudar – e muito – em sua utilização.

Comecei um inventário de dicas para uso do Google num fórum do Semiofon (aberto a colaborações: http://www.semiofon.org/modules/newbb/viewforum.php?forum=20 >> escolha visualizar as mensagens desde o início para ver todas as dicas).

O próprio Google fornece algumas informações sobre isso em http://www.google.com.br/help/basics.html. O assunto é tão relevante que uma busca pelas palavras "como fazer busca no google" (sem as aspas) retornou aproximadamente 958 mil links, a maioria para blogs, a maioria muito úteis.

Fica claro que quando alguém "simpaticamente" diz: "o google é seu amigo!", está fazendo um uso ingênuo ou sarcástico do pressuposto de que o destinatário "perguntante" conheça técnicas de busca que permitam utilizar com eficiência essa ferramenta de busca e retornar o resultado esperado (no lugar de milhares ou até milhões de resultados aproximados). Mais uma vez nos

deparamos com a possibilidade de que o destinador "informante" não esteja, de fato, muito preocupado em ajudar, mas, antes, em manter ou criar um status de hacker.

Por outro lado, esses artigos sobre como usar o sistema de busca nada mais são do que tutoriais. Um sistema de busca, como o Google, nada mais é do que um programa que precisa ser conhecido para ser bem usado. Como se pode ver nos tutoriais, o que é preciso aprender é uma sintaxe, muito afinada com a lógica de programação, portanto uma sintaxe não óbvia. Assim, um exemplo de resposta bem mais eficiente e afinada com a filosofia do software livre seria: "procure pelo erro colocando-o entre aspas no google, se alguém já teve o mesmo erro e resolveu você deve encontrá-lo". Afinal, esse método é usado muitas vezes pelos próprios destinadores "informantes" para buscar as respostas que dão em fóruns e chats; partilhar o método é ajudar o destinatário "perguntante" a ganhar independência.

Note que há muitas palavras-chave para compreender o sistema de comunicação implicado nessas relações. Voltaremos a elas na análise semiótica.

Definição informal de documentação

Uma das melhores definições que encontrei para documentação de software, no Google, foi:

> A **documentação de** *software* descreve cada parte do código-fonte, geralmente uma função, uma classe, um simples trecho ou módulo. Consiste também no conjunto de manuais gerais e técnicos, além de diagramas explicando o funcionamento de um *software* como um todo ou cada parte dele.
> (http://pt.wikipedia.org/wiki/DocumentaC3%A7%C3%A3o_de_software)

No entanto, essa definição pressupõe que você saiba o que é "software", "código-fonte", "função", "classe", "módulo". Note que somente três dessas palavras remetem a outras entradas na Wikipedia. Por isso buscamos aqui outra saída: partir da palavra "documentação" para compreender seu sentido.

A palavra documentação tem muitos sentidos diferentes, mesmo saindo do mundo do software. Por exemplo, para alugar um apartamento, você precisa ter disponível toda a documentação necessária, caso contrário não consegue alugar. Essa documentação é, portanto, o *conjunto de documentos* que provam que você tem condições de alugar (pagar) aquele imóvel. Mas não é só isso.

A documentação inclui documentos que provam que você é você mesmo, algo bastante esquisito e, muitas vezes, nos coloca em uma situação que pode se tornar bastante constrangedora,

tanto para quem exige quanto para aquele de quem se exige.

A documentação também inclui a própria lista de documentos necessários, algumas vezes indicando: rubrique todas as páginas e assine a última, cuja assinatura deve ser autenticada em cartório. Ou então dizendo em quantas vias a documentação deve ser entregue. Também explica como e quando deve ser feita a vistoria, que por si só já constitui um documento a mais para integrar a documentação do imóvel.

Além da parte à qual nós temos acesso imediato, a planta do imóvel é mais um documento que, junto com os registros na prefeitura e dos engenheiros, habite-se e outros documentos, constituem a documentação do imóvel que você está alugando.

Existem, portanto, três grupos de documentos que constituem essa documentação, todos eles encontrados no software:

a) uma parte dos documentos para alugar uma casa diz como e de que modo o imóvel foi construído (os próprios scripts do "imóvel", portanto os arquivos que compõem o programa, o código-fonte); estes documentos nem sempre estão acessíveis e muitas vezes não são compreensíveis pelo público leigo;

b) uma outra parte diz quem é o proprietário, quem aluga e sob quais condições (a "licença", no caso do software);

c) finalmente, uma parte explica como usar o imóvel (cláusulas do tipo “não coloque pregos nas paredes”, ou seja, os “tutoriais”, READMES e “manuais”, no caso de softwares).

Isso sem contar com o material de divulgação.

Para compreender o peso dessa documentação vamos partir de uma situação hipotética. Você entra no imóvel e pensa: “ora, claro que eu vou colocar um quadro na parede”. E, já que pregos são proibidos, você usa um parafuso.

A vistoria dizia que a pintura era nova. Na primeira chuva cai metade do reboco e já passou a fase da revisão de vistoria. Se fosse software livre, você relataria o bug (erro, problema), tranqüilamente, e possivelmente até encontrasse alguém num chat ou fórum que te apontasse uma solução rápida.

Mas seu imóvel não é livre. Você sabe que, sejamos francos, não há solução: o reboco caiu e

você vai ter que conviver com isso até ir embora (quando vai ser obrigado a entregar uma pintura nova) ou pinta e repinta por sua própria conta com a certeza de ter comprado (alugado) gato por lebre. O seu direito termina antes que você perceba.

Existem, claro, desenvolvedores/proprietários bastante amigáveis: você avisa sobre o problema e ele vai procurar a solução melhor para todos o mais rápido possível. No entanto, sejamos realistas, a maioria vai esconder o bug/problema, o problema é seu, afinal assinou a vistoria (assumiu o risco da tela azul de morte), e quando você sair e pintar tudo bonitinho como encontrou, ele aluga para outro usuário.

Aluga? Como assim? Software a gente compra, não aluga. Será? Quando eu compro uma maçã, tenho o direito de fazer o que quiser com ela, desde comê-la até fazer uma torta e vender. Com o software eu posso fazer isso? Depende da licença.

Vejamos outro exemplo: o livro que você compra é seu? Segundo o texto publicado no Slashdot sobre licenças (veja na underlinux: http://under-linux.org/8353-meu-livro-e-meu.html), não. O texto fala de livros digitais, vendidos online: será seu o livro de papel que está na sua prateleira de livros? Não. Ele também está sujeito a leis de direito autoral. Assim, eu comprei somente o direito de ler. Se eu quiser posso passar esse direito adiante, posso vender esse direito, mas não posso copiar o livro, reproduzir partes dele (coisa que na faculdade se faz todo dia: somos todos piratas!). As licenças restritivas seriam, então diferentes das licenças livres? Sim, porque a licença livre pode lhe permitir copiar, desde que citada a fonte. Mas e quando eu copio um trecho do livro de papel e cito a fonte? Não é a mesma coisa? Fazemos isso todo dia em trabalhos escolares.

É muito comum editoras comprarem direito à reprodução (especialmente em textos visuais ou sonoros, mas também de textos como letras de música) para fazer as tais citações.

Se a licença fosse Creative Commons (http://www.creativecommons.org.br/), não seria preciso comprar. A versão mais conhecida permite reproduzir, desde que citada a fonte. Estaria bem mais de acordo com a finalidade: não fazemos trabalhos científicos para vender, eles servem para partilhar o conhecimento. Mas a cultura geral e leiga (em termos de licenças) imagina que, se for livre, é de domínio público, não tem dono. E todo autor quer ter o direito de manter a autoria do seu trabalho. A isso, no meio do software livre, chama-se meritocracia: o trabalho deve ser reconhecido, portanto a autoria deve ser mantida e evidenciada.

Cabe, portanto, compreender melhor a diferença entre uma licença livre e domínio público antes de prosseguir a análise.

Licença livre (de código aberto) não é domínio público

A questão foi discutida no artigo "Is public domain software open source?" (http://www.news.com/8301-13580_3-9881858-39.html), publicado por Stephen Shankland. É interessante pensar a questão não só do ponto de vista do software livre, mas da cultura livre em geral. Por exemplo: o que ganha um autor que quer viver de direitos autorais e publica seus artigos com a Creative Commons?

A questão dos direitos autorais mudou muito na história. Vamos entrar numa história pessoal para entender do que se trata. Na minha experiência musical, eu era uma compositora (veja exemplos em http://www.butiazal.com/acris/) extremamente preocupada com direitos autorais, ou seja, com muito receio de que as músicas por mim compostas pudessem ser eventualmente plagiadas. Afinal, a música era meu trabalho, como podia aceitar alguém plagiá-la e ganhar com ela em meu lugar?

Dois eventos durante meu curso de graduação em música popular, na Unicamp, fazem muito sentido hoje, quando penso em licenças.

O primeiro evento foi um comentário feito em aula. Segundo o prof. Roberto Zan, numa alusão a Dostoyevski, se não me engano, o artista, para viver a nova era da música comercial, deveria deixar a aura na sarjeta, por mais que desejasse pegá-la de volta. Fazer isso, retomar a aura, seria retomar uma aura suja, não mais a antiga aura dos artistas próximos dos deuses. Artista hoje trabalha, comercializa sua arte para sobreviver, sem o apadrinhamento dos nobres medievais e renascentistas.

O segundo evento foi mostrar, para o professor Gogô (pianista) uma música que eu acabara de compor, um samba em 5/4, ritmo bastante incomum nos nossos dias. Ele gostou, mas disse: olha, isso não é exatamente uma novidade, já se fazia samba assim nos anos 30... Em música, ele continuou, nada é novo, tudo é construído em cima de algo que já existia antes. Fiquei profundamente decepcionada, na época... lá se foi minha pobre e suja aura de volta para a sarjeta. E eram uma vez os deuses artistas.

De fato, o conhecimento humano em todas as áreas é sempre uma reconstrução, uma renovação de algo que já foi, um reinício a partir de algo que já existe.

O direito autoral surgiu como um direito comercial: se eu quiser gravar uma música de outra pessoa, preciso pagar direitos autorais, caso ela esteja licenciada com uma licença copyright. Se for de domínio público, caem todas as licenças e todo uso é absolutamente livre. Uma música

licenciada em copyright só cai em domínio público 50 anos após a morte do autor, ficando os direitos para sua família enquanto isso não acontece.

Quando cair em domínio público, ela pode ser usada como for, ninguém pode questionar.

No entanto, se eu libero uma canção em Creative Commons, o primeiro direito, e talvez o mais importante, é garantir a autoria. Todos podem copiar, gravar, fazer arranjos encima, mas a licença exige que seja mantida a autoria. Seria a volta da aura? Não. Simplesmente garanto que meu trabalho é meu trabalho, sem impedir que o resultado desse trabalho seja usado por outros que o desejarem. Nada mais delicioso que ouvir na voz de outra pessoa uma música que você compôs. Você vai cobrar por esse prazer?

No software é idêntico. É possível ganhar muito dinheiro com música ou com software. Quanto mais gente ouvir sua música e gostar, maior seu poder de fogo no mercado. Quanto mais gente disser que seu software é bom, idem.

Mas código aberto ou Creative Commons não são domínio público. Domínio público é terra sem lei.

Será que os usuários têm medo disso quando preferem um código fechado a um código aberto? Será que pensam que, como o código é aberto, estamos numa terra sem lei?

A situação é totalmente inversa. Se você usa um programa fechado, somente as pessoas que o fizeram sabem exatamente o que esse programa faz. Afinal, todo programa possui processos visíveis e processos invisíveis, isso é natural, mas, se o dito programador quiser, numa atitude bastante ilícita, pode muito bem incluir entre os processos invisíveis e necessários um único que envie dados sobre sua máquina, seu IP, suas senhas ou seus links prediletos enviando esses dados sabe-se lá para onde. E ninguém ficará sabendo. Nem seu anti-vírus. É por esse motivo que, cada vez mais, os bancos, por exemplo, buscam soluções livres.

Claro, essa “atitude ilícita” é crime passível de processo. Mas, enquanto isso não acontece (se acontecer), você está desprotegido e nem mesmo sabe disso.

No software livre de código aberto é exatamente o contrário. Qualquer programador do mundo (ou quem mais quiser) pode ver o que seu programa está fazendo, tanto os processos visíveis quanto os invisíveis, pois código aberto significa código publicado, acessível.

Durante um trabalho de tradução numa comunidade de software livre até eu, que na época não programava nada, fui capaz de descobrir num tema de portal web distribuído livremente nessa comunidade um elemento estranho: uma propaganda que aparecia no site sem que os usuários finais fossem capazes de removê-la. Mas como o código era livre, foi possível descobrir onde estava o

elemento ilícito, corrigir e divulgar uma versão sem esse elemento. Além disso, a divulgação da atitude ilícita do autor da propaganda impediu que ele repetisse o ato, por receio de ser excluído da comunidade.

Era uma simples propaganda, poderia ser algo bem pior como envio de dados cadastrais. Se alguém quiser tentar uma falcatrua dessas no software livre, sabe que sua carreira acaba. Principalmente porque, no software livre, a primeira garantia são os créditos, ou seja, a meritocracia da qual já falamos. Quem vai dar credibilidade a alguém que já jogou sujo uma vez?

Portanto, não estamos numa terra sem lei quando usamos software livre: estamos numa terra com leis bem claras e, além disso, com muitos vigias e advogados bem intencionados. No Brasil, inclusive, um artigo publicado sob Creative Commons (como, por exemplo, na revista Texto Livre, http://www.textolivre.net) garante juridicamente a autoria para seu autor.

Software livre de código aberto não é, de forma alguma, domínio público. Mas o que, então significa essa tal liberdade? Liberdade vigiada?

Documentação é liberdade? Análise semiótica

Antes de tentar responder à questão que fechou o tópico anterior, vale a pena "abrir o código", ou seja, explicitar a análise semiótica que está sendo desenvolvida aqui.

Meritocracia, liberdade e conhecimento são, por assim dizer, as palavras-chave dessa discussão. Vamos pensar essas palavras-chave do ponto de vista da semiótica tensiva (FONTANILLE & ZILBERBERG, 2001).

O *conhecimento* é um objeto compartilhável: quando um destinador "informante" doa algum conhecimento para o destinatário "perguntante", o destinador "informante" continua possuindo esse conhecimento compartilhado.

Compartilhar vs. Restringir constituem uma profundidade extensa: compartilhar significa espalhar, enquanto restringir significa concentrar; compartilhar: tornar público; restringir: tornar privado. À máxima concentração corresponde a menor extensidade, o menor "espalhamento". A profundidade intensa corresponde a uma categoria de controle: o maior ou menor controle ganham sentidos diferentes (valorização oposta) conforme o quadro de valores no qual se insere.

Expliquemos: um objeto compartilhável pode ser compartilhado ou não. Existem basicamente dois quadros de valores, caracterizados por sistemas tensivos opostos, que definem a validade social do ato de compartilhar ou restringir um determinado objeto:

a) compartilhamento negativo: bastante conhecido na sociedade capitalista, na qual segredos de produção são comuns e incentivados. Nesse quadro de valores o máximo controle é obtido com o mínimo de espalhamento da informação. Temos um sistema tensivo como na ilustração 1. O controle tem valor positivo, portanto quanto maior o compartilhamento, menor o controle, menor o valor de mercado, maior o peso negativo da escolha.

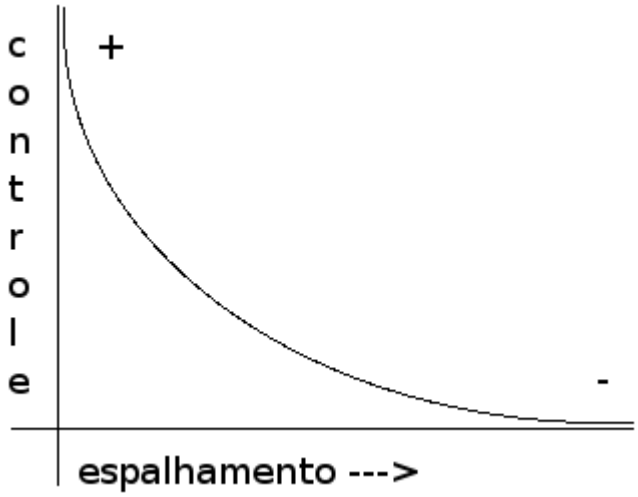

Ilustração 1: Neste esquema tensivo inverso quanto maior o compartilhamento da informação ou do conhecimento, menor o controle sobre o objeto. O controle tem valor claramente positivo.

b) compartilhamento positivo: por exemplo na filosofia do software livre, mas também na educação, para a qual compartilhar conhecimento é sua essência mesma. Aqui o controle muda de sentido: deixa de ser privado para ser público; por exemplo, um resultado de pesquisa que não for publicado está sujeito a plágio sem direito de reclamação, enquanto o plágio de um artigo publicado é considerado crime. Assim, a publicação garante o controle sobre os resultados. O controle continua sendo valorizado positivamente, mas o sistema tensivo é oposto ao anterior, como se pode ver na ilustração 2.

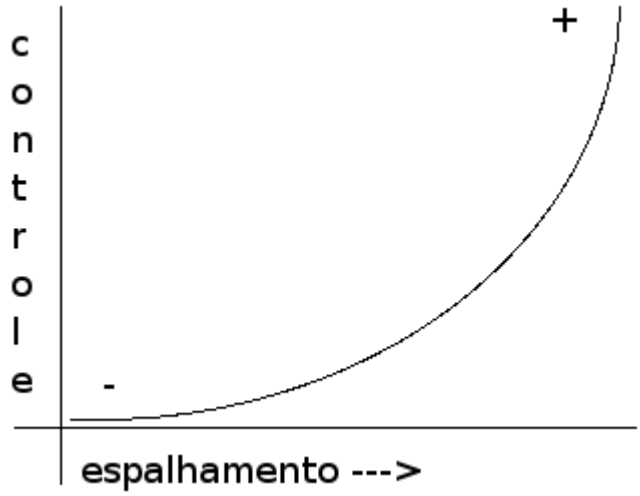

Ilustração 2: Aqui o esquema tensivo é converso: quanto maior o compartilhamento da informação ou do conhecimento, maior o controle sobre o objeto. O controle continua tendo valor positivo.

Cabe agora pensar tensivamente a *meritocracia*. Mérito: valor obtido pelo fazer. Na meritocratica, o fazer tem valor positivo. No caso, por exemplo, do software proprietário, o dono não é necessariamente quem fez, mas para quem fez: para uma empresa ou para si mesmo. Assim, podemos dizer que os dois sistemas tensivos acima sobremodalizam o quadro de valores da meritocracia, pois o espalhamento da informação (ou, porque não?, do *conhecimento*) sobre a autoria é fator essencial para a própria existência da meritocracia. O máximo controle novamente é positivo.

No software livre, mantemos a relação conversa (como na ilustração 2) entre o espalhamento e o controle: quanto maior o espalhamento da informação sobre o autor, maior o controle sobre a autoria. Já no caso do software proprietário o controle não diz respeito à autoria, mas à propriedade: quanto mais concentrada (menor o espalhamento), maior o controle sobre ela. Observamos que em todos esses casos o controle tem valor positivo.

Finalmente chegamos à *liberdade*. Continuemos opondo os dois sistemas de compartilhar ou restringir (*conhecimento*). Em todas as situações observadas, o *máximo controle* tinha valor positivo (o valor positivo aumenta com o aumento do controle). Numa primeira análise a liberdade teria, portanto, valor negativo, pois o controle seria o oposto de liberdade. Os fundamentos do software livre e do software proprietário indicam, no entanto, uma leitura bem diversa. No caso do software proprietário, a liberdade consiste no /poder/ ter acesso a qualquer programa bastando para isso comprá-lo, ou seja, liberdade é questão de /querer/ e /poder/ (aquisitivo).

No site da FreeSoftware Foundation América Latina temos a seguinte definição de software livre:

> "Software Livre é uma questão de **liberdade**, não de preço.
> Software Livre respeita quatro liberdades essenciais:
> 0. de rodar o programa quando quiser;
> 1. de estudar o código fonte e modificá-lo para que faça o que você quiser;
> 2. de copiar o programa e distribuir as cópias quando quiser;
> 3. de publicar ou distribuir uma versão modificada quando quiser." (http://www.fsfla.org/svnwiki/)

No caso do software livre, a liberdade é questão de /dever/ e /poder/: eu não só posso compartilhar, copiar, modificar, usar o software como quiser como também, e principalmente, devo garantir essas liberdades. A liberdade deixa de ser um valor absoluto: são várias e específicas liberdades. A própria liberdade passa a ser um fator de controle contra a privatização do software, como pode ser notado ainda no texto de apresentação da página da FSL LA:

> Nossa missão é *defender os direitos e as liberdades de usuários e desenvolvedores de software*, lutar pela tua liberdade de *executar* o software que uses *para qualquer propósito* que queiras, de *estudar seu código fonte* se quiseres e *adaptá-lo*, para que faça o que queiras, e de *copiá-lo, distribuí-lo e publicá-lo* quando quiseres, *com ou sem as melhorias* que tenhas feito. Dessa forma, tu, nós e todos poderemos *usar computadores em liberdade*.
> Quando permites que outros desrespeitem tuas liberdades sobre algum software que uses, o dano não se limita a ti. Sem resistência, vão estender seu poder sobre ti e todos.
> É um *problema social*, curável com tua ajuda, *rejeitando o Software não-Livre e substituindo-o por Livre*. Batalha por tuas liberdades: tenta resistir ao controle imposto sobre nós através do software que usamos.
>
> **Quanto mais gente resistir,**
> **mais gente será Livre, e**
> **mais gente será livre para ser Livre.**
> **Para teu próprio bem e**
> **em solidariedade a todos,**
> **escolhe a liberdade.**
> **Sê Livre!"**
>
> (http://www.fsfla.org/svnwiki/) (negrito original)

Mais uma vez o espalhamento é tomado como uma variável de controle e este, por sua vez, tem valor positivo. O software livre, portanto, trata a liberdade como trata o conhecimento, num sistema tensivo convergente: quanto maior o espalhamento, maior o controle. Já o software proprietário trata a liberdade como uma variável controlada pelo poder aquisitivo: o controle ainda tem valor positivo, mas num sistema inverso; assim, quanto maior o espalhamento, menor o valor da liberdade (já que a liberdade é uma questão de poder aquisitivo nesse quadro de valores, ela é uma variável quantitativa).

Documentação é liberdade

A documentação é o conjunto de documentos que torna um código aberto acessível aos interessados: são documentos que explicam como funciona, como foi feito, como pode ser usado e o que foi feito em cada atualização. Documentação, portanto, é a ferramenta de acesso ao *conhecimento*. É, portanto, uma ferramenta de espalhamento. Como vimos acima, o máximo espalhamento no software proprietário é negativo, enquanto no software livre é positivo (ver ilustrações 1 e 2).

Isso explica por que a documentação disponível para o público, no caso do software proprietário, restringe-se ao "como usar", enquanto no caso do software livre trata-se de toda a documentação de software produzida (como funciona, como usar etc).

Podemos concluir, portanto, que documentar é viabilizar o espalhamento, de modo que documentação é, com certeza, liberdade. É interessante pensar nos exemplos acima, da composição de canções às licenças de livros e artigos: embora a documentação nesses casos esteja muito mais ligada à divulgação e ao acesso do que ao modo de produção, também aqui documentar é tornar livre.

O modo como o conceito de liberdade é tratado nas duas perspectivas analisadas é completamente diferente: a liberdade vs. as liberdades. Ao generalizar a liberdade, o sistema capitalista dificulta sua definição, permitindo a acepção até mesmo fantasiosa de uma possível liberdade absoluta e infinita. Essa acepção é bastante divulgada pela mídia, mostrando que nossa cultura resiste a um racionalismo que aceitaria a liberdade como múltipla e específica, limitada e controlável. Observe na ilustração 2 esse limite: o espalhamento tem um limite, ao contrário da ilustração 1 na qual o espalhamento tende ao infinito. Isso explica o caráter político da apresentação da FSF América Latina e indica que o software livre está plenamente consciente de que as mudanças por ele propostas vão muito além de um modismo: trata-se de uma mudança de paradigmas sociais e culturais que só pode realizar-se a médio ou longo prazo.

Referências Bibliográficas

Aprenda o básico sobre o Google: <http://www.google.com.br/help/basics.html>.
BARROS, Diana Luz Pessoa de. A comunicação humana. In: *Introdução à Lingüística*/José Luiz Fiorin (org.). – São Paulo: Contexto, 2002.
Classe: <http://wapedia.mobi/pt/Classe>.
Creative Commons.org <http://www.creativecommons.org.br/>;
Documentação de software:

<http://pt.wikipedia.org/wiki/DocumentaC3%A7%C3%A3o_de_software>
FONTANILLE, Jacques & ZILBERBERG, Claude. *Tensão e Significação.* São Paulo: Discurso Editorial: Humanitas/FFLCH/USP, 2001.
Fórum com dicas sobre Google. <http://www.semiofon.org/modules/newbb/viewforum.php?forum=20>
FreeSoftware Foundation América Latina: <http://www.fsfla.org/svnwiki/>
Função: <http://wapedia.mobi/pt/Função>
Meu livro é meu? <http://under-linux.org/8353-meu-livro-e-meu.html>
Músicas de Ana Cristina Fricke Matte: <http://www.butiazal.com/acris/>
O que é software livre? <http://www.fsfla.org/svnwiki/about/what-is-free-software>
Revista Texto Livre: Linguagem e Tecnologia: <http://www.textolivre.net>
Software: <http://wapedia.mobi/pt/Software>
Stephen Shankland. Is public domain software open source? <http://www.news.com/8301-13580_3-9881858-39.html>
Wikipedia: Eduardo Henrique Rivelli Pazos <http://pt.wikipedia.org/wiki/Usuário:Eduardo_Henrique_Rivelli_Pazos>
Wikipedia: RTFM. <http://pt.wikipedia.org/wiki/Rtfm>

Obs: todos os links foram acessados em 15 de julho de 2008.