UNIVERSIDADE FEDERAL DE SANTA CATARINA

Open Source, criação e música: uma proposta de aproximação entre o desenvolvimento de software aberto e a criação musical.

Martim Azevedo do Nascimento

UNIVERSIDADE FEDERAL DE SANTA CATARINA
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA
CURSO DE SISTEMAS DE INFORMAÇÂO

Open Source, criação e música: uma proposta de aproximação entre o desenvolvimento de software aberto e a criação musical.

Martim Azevedo do Nascimento

Trabalho de conclusão de curso apresentado
como parte dos requisitos para obtenção do
grau de Bacharel em Sistemas de Informação

Florianópolis - SC

Martim Azevedo do Nascimento

Open Source, criação e música: uma proposta de aproximação entre o desenvolvimento de software aberto e a criação musical.

Trabalho de conclusão de curso
apresentado como parte dos
requisitos para obtenção do grau
de Bacharel em Sistemas de Informação

Orientador: Roberto Willrich

Banca examinadora:
Vitório Bruno Mazzola
Rosvelter Coelho da Costa

RESUMO

O presente trabalho propõe uma aproximação dos conceitos de produção colaborativa vivenciados pelos projetos de software aberto para um domínio musical. Através de pesquisa bibliográfica são analisados os diferentes fatores que levaram a este contexto, ao qual o open source se insere, de ubiquidade das redes e produção social baseada em commons, focando nas mudanças de paradigmas na função do autor e no ato criativo em diferentes épocas da história. O trabalho explora os mecanismos internos técnicos e psicológicos do open source, tentando uma generalização para a aplicação em outros domínios, como a arte, e em especial a música. Por fim, apresenta o design e a implementação do Hermeto, uma plataforma de colaboração musical inspirada nos conceitos extraídos pelo open source, como o uso de task tracking, controle de versões e wikis.

ABSTRACT

This paper proposes an approximation of the concepts of collaborative production, experienced by open source software projects, for a musical domain. Analyzes the different factors that led to this context, to which open source is inserted, the ubiquity of networks and commons based peer production. The work explores the inner workings of open source psychological aspects and technical infrastructure, trying to achieve widespread application in other fields such as music. Finally, presents the design and implementation of Hermeto, a musical collaboration platform inspired by patterns from the open source world, such as the use of task tracking, version control, and wikis.

SUMÁRIO

Listagem de figuras

1 INTRODUÇÃO

O ser humano é um ser essencialmente social. Grande parte de suas ações estão de alguma forma ligadas ao bem comum. Mesmo quando uma atitude parece ter suas origens enraizadas no ego, com princípios individualistas, mesmo assim somente o faz porque existe uma sociedade a qual conhece e a qual suas lembranças irão sempre o influenciar.

A arte, por sua vez, surge como experiência afirmativa desta necessária socialização ao demostrar sua potencialidade como instrumento de disseminação do que é considerado belo para um povo ou grupo de pessoas. Desde sua origem sua função primordial foi comunicar à eternidade o ponto de vista de uma época representada pelos seus indivíduos na forma da obra de arte. Seja consolidando ou questionando a tradição, a figura do autor aparece como ser atuante na atividade criativa, com sua importância como personagem principal variando ao longo da história em maior ou menor intensidade.

Por fim, a tecnologia aparece como fator modificante das interações sociais e como consequência, das ações criativas que surgem destas interações. O surgimento da internet potencializou a emergência de movimentos de criação colaborativa como o open source e as wikis. A arte, no entanto, ainda não recebeu a devida atenção nestas novas plataformas de colaboração.

O presente trabalho propõe uma aproximação dos conceitos vivenciados pelo movimento open source para o mundo da música através do desenvolvimento de uma plataforma de colaboração musical. Além disso pretende analisar a atividade criativa e a função do autor sob o prisma da tecnologia e procura demonstrar que a mesma teve papel fundamental nesta mudança de paradigma.

No primeiro capítulo propõe-se uma breve sistematização do papel do autor ao longo da história, demonstrando as mudanças de paradigmas que o ato da criação sofreu em diferentes épocas, apresentando as motivações do autor e sua importância para o processo criativo.

O segundo capítulo aproxima esta sistematização no momento atual, dissertando sobre os fatores que levaram a humanidade a este paradigma de criação colaborativa e ubiquidade das redes, dentro de um contexto de uma economia de informação conectada.

O terceiro e quarto capítulo foca nos movimentos de desenvolvimento colaborativo de software e busca entender os mecanismos por trás destes movimentos, tentando uma generalização destes mecanismos através da procura de semelhanças com outras atividades criativas, como a música.

Por fim, o quarto capítulo apresenta o Hermeto, um ambiente colaborativo para composição musical, sua arquitetura, as motivações por trás de seu design e as tecnologias utilizadas.

1.1 OBJETIVOS GERAIS

- Fazer um estudo sobre os processos criativos no desenvolvimento de software aberto
- Propor uma aproximação entre os mecanismos de colaboração utilizados nos projetos open source para um contexto musical

2 A FUNÇÃO DO AUTOR AO LONGO DA HISTÓRIA

É notável como a humanidade modificou sua visão de arte e criação como atividade individual ao longo do tempo, oscilando entre a idéia de uma criação nascida da tradição e tendo o coletivo anônimo como principal autor para uma visão aonde a genialidade individual do criador aparece como fator primordial.

O objetivo deste capítulo é contextualizar historicamente os diferentes momentos vivenciados pela humanidade no que tange a criação e à produção de conhecimento.

Utilizando a divisão da história proposta por Lemos (Lemos, 2002) e consolidada por Primo (Primo, 2008) tenta-se sistematizar esta mudança de perspectiva sobre o papel do autor. Esta divisão foi escolhida por fazer uma análise dos diferentes processos de conhecimento vivenciados pelo homem através da ótica do desenvolvimento tecnológico, o que nos aproxima ao tema deste trabalho.

Antes, no entanto, iniciamos nossa discussão sobre a história da autoria através de uma observação biológica sobre a origem da coletividade e vida em sociedade, a qual irá nos guiar em alguns momentos ao longo do trabalho.

Segundo Wright (Wright, 2007), ainda nos primórdios da vida na terra com o surgimento de organismos mais complexos com alguns trilhões de células como insetos e pássaros e órgãos mais especializados como olhos e células nervosas uma maior autonomia e independência foram evoluções naturais. No entanto tal independência veio com o preço de um isolamento maior entre os seres. A necessidade de se entender levou então os primeiros seres a uma interação baseada principalmente na tentativa de imitar um ao outro. A imitação surge então como "a sinapse do cerébro social" ao permitir que a comunicação pudesse ser estabelecida e a experiência coletiva pudesse ser compartilhada novamente. "A imitação surge como uma cola que une estes cada vez mais autônomos seres vivos dentro de uma mais elevada ordem de organização: as redes sociais." (Wright, 2007)

2.1 TRÊS FASES DO DESENVOLVIMENTO TECNOLÓGICO

O desenvolvimento tecnológico esteve sempre diretamente ligado à intensidade em que os processos de conhecimento se apresentaram para o homem, sejam estes a educação, criação e disseminação. Segundo Lemos (Lemos, 2002) podemos dividir a história em três níveis de desenvolvimento tecnológico: a fase da indiferença, a fase do conforto e a fase da ubiqüidade.

2.1.1 Fase da indiferença

A fase da indiferença se apresenta como a mistura entre arte, religião, ciência e mito. O limite entre o mágico e a realidade é extremamente tênue. Nesta fase o conhecimento é um dom concedido por Deus. A tradição é um peso que se apresenta em todas as formas de expressão, onde a criação nada mais é do que a repetição pouco modificada do saber herdado ao longo da história e creditado a um Deus. Na fase da indiferença o artista é uma pessoa autorizada a ouvir a voz divina. Segundo (Primo, 2008),

> *"(...) Mesmo que essas narrativas sofram modificações com o tempo, que outras novas sejam "reveladas" por pessoas autorizadas a ouvir a voz divina, elas visam dar sentido ao desconhecido, criam uma moral e suas punições informam a vida. Sendo o saber uma inspiração (ou imposição) divina, a própria concepção de autoria decorre dessa visão de mundo. Os textos do período raramente são acompanhados do nome do autor." (p.54)*

A cópia nesta fase é uma atividade mais que permitida, necessária. Em um processo semelhante ao vivenciado pelos primeiros organismos, a experiência coletiva é vivenciada através da imitação e cópia. A falta de mecanismos efetivos de disseminação e perpetuação do conhecimento tornaram a oralidade e os rituais os únicos meios de transmissão de uma época. Mesmo com o desenvolvimento da escrita a reprodução de textos dependia da cópia manual por escribas.

A arte possui papel importante nesse contexto propagando as tradições e funcionando como relato histórico. Até a invenção do livro e da imprensa os contos e canções folclóricas bem como pinturas e outras artes serviram como os únicos meios de transmissão da tradição entre gerações.

Na música, os engravers, pessoas responsáveis por transcrever as partituras para o papel, muitas vezes com procedimentos demorados e artesanais, serviam como tradutores da arte e do conhecimento de sua época para as futuras gerações.

2.1.2 Fase do conforto

A fase do conforto é determinada pela razão iluminista. Em contraste às trevas da idade média, o período é marcado pela separação entre ciência e religião. Neste momento a razão surge como estandarte em todos os processos de criação e conhecimento.

Descartes define uma filosofia que terá reflexos nos mais diversos meios de criação e produção

de conhecimento. Sua visão mecanicista do mundo prega a divisão da mente e da matéria, sendo que cada entidade no universo poderia ser classificado em um desses domínios. No domínio da matéria ainda caberiam mais divisões, aonde, através da metáfora da máquina, seria possível analisar e reproduzir cada entidade material como pedaços de um todo maior.

Esta visão irá influenciar as mais diversas áreas do conhecimento, tendo reflexos imediatos em disciplínas como física, química e biologia e sendo a base para os sistemas educacionais como conhecemos hoje.

É também inspirado por Descartes que a humanidade passa a questionar certos dogmas intocáveis da primeira fase. Segundo Primo,

"(...) a modernidade passa a abordar o conhecimento como um processo que deve ser desenvolvido a partir da dúvida eterna. Se até então a tradição não deveria ser desafiada, a partir de agora a verdade deve ser buscada desde a análise sistemática e metódica dos fenômenos. (...) Ciência e progresso vinculam-se. Antes, as determinações da natureza deveriam ser obedecidas. A eventual cólera da natureza só poderia ser explicada pela vontade e insatisfação das forças do além. O homem moderno, por sua vez, passa a estudar a natureza com o fim de dominá-la." (Primo, 2008)

Este pensamento dominador vai ter papel influente no desenvolvimento científico e tecnólogico. A ordem neste momento é desafiar a natureza, procurando a todo custo descobrir os mecanismos que a regem. E se em um primeiro momento esta busca incessante estava dentro da idéia romântica de ciência, ela também irá servir perfeitamente aos interesses do capital quando este se desenvolve como sistema de produção.

É a época do taylorismo e do fordismo. A produção de bens de consumo se torna o carro-chefe do capitalismo. Os bens de consumo para serem rentáveis precisam ser raros e deprecáveis. A necessidade das pessoas é o maior impulsionador do consumo e como veremos mais a frente, esta necessidade precisa se tornar facilmente inventada. É o surgimento do marketing.

A arte e música também são influenciadas por este pensamento através do processo de rarificação do autor e da rivalização da obra de arte. Neste processo o autor é transformado em um ser mítico que tem o poder de transformar a natureza e criar coisas belas. Ele é raro, porque a lógica de oferta/procura do consumo precisa que seja. Segundo esta lógica, não é qualquer um que possui as habilidades necessárias para fazer arte. Somente aqueles que interessam a indústria, normalmente com os quais mantêm contratos e para os quais investem na criação de uma imagem mítica, têm o direito de transgredir e criar.

O nome do autor assinando a obra tem, nesta fase, significados muito mais profundos do que nas

obras não-anônimas da primeira fase. Ao assinar a obra o autor passa a ser responsável por aquilo que está sendo expressado. A responsabilidade individual torna o autor passível de julgamento pela sociedade e seu discurso passa a ser julgado não simplesmente pelo que está sendo dito, mas sim por tudo que envolve o nome do autor, seu contexto histórico, sua história de vida e seu currículo artístico. Alex Primo, citando Foucalt (Foucault, 1992, p. 46) revela:

"A inclusão do nome do autor em uma obra particular passa a indicar, (...), que aquilo não se trata de um discurso qualquer, cotidiano e passageiro, mas de um discurso que precisa ser abordado de uma dada maneira, dentro de uma certa cultura." (Primo, 2008)

A rivalização da obra de arte se dá nesta mesma lógica do consumo. Segundo (Simon, Vieira, 2008),

"(...) é rival aquele bem ou recurso cujo uso por alguém impede (ou compete com) o uso por outra pessoa."

A rivalização está diretamente ligada à escassez ou abundância de determinado recurso. Ninguém paga para respirar, mas a água potável é um bem rival, devido à sua escassez.

A obra de arte é por natureza um bem não-rival. E é assim porque tem como produto final a informação, nesse caso informação estética. A informação, por si só, é considerada um bem intangível e não-rival . A rivalização da mesma se dá em uma lógica de mercado, onde arbitrariamente é capturada, empacotada e sinteticamente comercializada como algo rival.

2.1.3 Fase da ubiqüidade

A última das fases do desenvolvimento tecnológico citada por Lemos é o momento presente, representado pela ubiquidade e pelas redes de informação.

Segundo Sérgio Ximenes (Ximenes, 2001) é ubíquo aquele que "está em todos os lugares ao mesmo tempo; onipresente". Em um contexto tecnológico, ubiquidade representa o fenômeno das tecnologias de informação conectadas, presentes nos mais diversos níveis dentro da sociedade, desde celulares até automóveis, computadores de bolso e lan-houses.

A onipresença se dá pelo fato dessas tecnologias permitirem a sensação de se estar em diversos lugares ao mesmo tempo. Através da internet é possível se conectar com o mundo em diversos

níveis em qualquer lugar e sem nenhum deslocamento físico.

Se na segunda fase a ênfase é no futuro, aqui o presente ganha importância através da urgência em se sentir participante do seu tempo, onde cada ação individual pode ter um alcançe realmente global.

Maffesoli (Maffesoli, 2006 in Primo, 2008) relata que, ao contrário da individualidade moderna da segunda fase, o indivíduo aqui é caracterizado pela importância que dá à sua identificação no grupo, como ser atuante no meio, mas hedonista no sentido de buscar sempre suas realizações pessoais. É o individual no coletivo, como sugere o slogan do concretista Helio Oiticica. Como iremos ver mais adiante esta característica irá explicar a motivação do indivíduo na participação de um ambiente de produção coletiva, como é o caso do open source.

Nesse sentido, "*(...) quer-se "(...) vibrar em comum, sentir em uníssono, experimentar coletivamente, tudo o que permite a cada um, movido pelo ideal comunitário, de sentir-se daqui e em casa neste mundo*". " (Maffesoli, 2006, p. 8, in Primo, 2008)

A visão de um gênio individual, tão importante para a indústria do entretenimento na segunda fase, é reavaliada neste contexto. Influencida principalmente pelas possibilidades de participação coletiva e alcançe de ações individuais, a criação e produção de conhecimento nesta terceira fase é cada vez mais democrática, permitindo uma visibilidade maior para os autores independentes, marginalizados até então.

Isso irá permitir toda uma nova geração de criadores nas mais diversas áreas, desde escritores autônomos, produtores de cinema, e músicos independentes até comediantes, jornalistas e programadores de software. Juntos ou individualmente irão transformar a maneira de enxergar o processo de criação e produção de conhecimento.

A própria noção de autor como se conhece tradicionalmente é modificada. Se ainda é possível a mitificação do autor como ser incompreendido e genioso como sustentador decadente de uma cultura do ídolo, da mesma forma que ainda é possível a criação anônima/mediûnica fruto de uma experiência divina hoje o que se vê é um hibridismo destes dois extremo: com uma reaproximação cada vez maior à função social do autor, com a preocupação de uma criação fruto da experiência social, mais sem o anonimato religioso e nem o aparato dogmático de épocas como a idade média.

Pelo contrário e o que o presente trabalho pretende mostrar é que as pessoas criam para si mesmas, ou como moeda de troca dentro da sociedade em que se incluem a fim de conseguir benefícios ou como meio de interação dentro de seu contexto social.

3 ECONOMIA DE INFORMAÇÃO CONECTADA

No primeiro capítulo mostrou-se uma revisão histórica dos diferentes momentos em que a tecnologia se apresentou como fator modificante do status quo e sua relação com a produção de conhecimento e a função do autor. Neste segundo capítulo será dado um enfoque especial para esta terceira fase, a qual Lemos demominou de fase da ubiquidade e como veremos, Benkler(2006) denominou de 'networked information economy', ou no português e como será chamada no restante do trabalho, economia de informação conectada.

O objetivo deste capítulo é demonstrar quais foram os fatores que levaram a este contexto e que papel fundamental teve a tecnologia e principalmente o movimento open source desde seus primórdios com a Free Software Foundation, Linux, FreeBSD e nas universidades para que este momento pudesse ser estabelecido.

Para isso procurou-se embasamento principalmente no trabalho de Yochai Benkler para alguma contextualização política e econômica além de buscar nos primórdios da história da internet e do open source alguns fatores importantes para o estabelecimento de uma rede realmente democrática e emergente em detrimento de uma rede autoritária e imposta.

3.1 CARACTERÍSTICAS

Vive-se um momento de euforia entre intelectuais e interessados. Pela primeira vez na história se tornou possível a construção de um ambiente participativo de produção de conhecimento que permite e incentiva a prática da colaboração como estratégia para o aprofundamento das liberdades individuais e para a construção de uma economia baseada na produção social de bens.

Hoje, mais do que nunca, o indivíduo comum tem a possibilidade de participar ativamente da economia global, não mais como um mero espectador ou apenas atrás de uma grande empresa como um empregado, mas sim com poderes de modificar e influenciar a produção de bens e conhecimento. Um jornalista independente no Oriente Médio tem a oportunidade de publicar uma notícia que pode ser lida por qualquer pessoa no México, assim como uma empresa de estudantes recém-formados no sul do Brasil pode competir com grandes multinacionais americanas.

Segundo Benkler (2006, p15) foram dois os principais fatores responsáveis por essa nova economia. O primeiro grande fator foi o desenvolvimento, de um século para cá, de uma grande indústria da informação, onde os bens são intangíveis, não-rivais e facilmente digitalizados.

O outro grande fator foi o surgimento de um ambiente de comunicação construído em cima da idéia de crescimento autônomo e descentralizado, onde com um computador de baixo custo

qualquer pessoa pode se conectar e participar de uma rede com alcançe global como é o caso da internet.

Benkler afirma que tal economia se caracteriza principalmente pela possibilidade de ações individuais e descentralizadas, com intenções não necessariamente comerciais, modificarem efetivamente o meio global, participando ativamente de uma economia cada vez mais democrática.

Segundo ele são três fatores que diferenciam esta nova forma de produção de uma forma de produção tradicional. Primeiro é que em uma economia onde se produzem bens de informação e onde a integração entre os sistemas de informação são efetivamente o grande diferencial entre estar ou não participando da rede, são os padrões abertos que entram como alternativas naturais em detrimento às alternativas proprietárias. Tal observação já não se aplica em uma indústria tradicional, onde os segredos de negócio e as patentes são altamente valorizados.

A segunda diferença é a já citada descentralização dos atores participantes do processo produtivo. Qualquer pessoa conectada à internet está equipada com o material necessário para produzir nesta nova economia.

Finalmente a terceira diferença, a qual está intimamente ligada com este trabalho, é o surgimento das plataformas de produção colaborativa, ou peer production, como as comunidades open source e a própria Wikipédia. Tais plataformas demonstraram ser possível a existência de uma economia participativa, onde quem dita as regras não são realmente as grandes corporações ou governos e sim os próprios participantes, organizados em uma hierarquia baseada no mérito, qual seja, aqueles que efetivamente colaboram para o bem da comunidade possuem certos direitos em relação ao grupo.

Neste ambiente de produção colaborativa o que mais chama a atenção é a democratização no acesso e participação. A hierarquia citada acima, não é de nenhuma maneira excludente. De fato qualquer pessoa que queira editar a Wikipédia pode o fazer. Qualquer um que queira fazer alguma modificação em um software open source também o pode. A diferença é que quem ganha o direito de colocar este código na base de código principal é aquele que fizer uma modificação coerente e de qualidade e somente depois de passar por uma revisão de algum contribuidor mais antigo.

3.2 PRODUÇÃO SOCIAL E OS COMMONS

Commons, na visão de Benkler, se identifica como sendo o oposto à propriedade. Em um contexto de propriedades, a lei determina quem é o dono da mesma e este decidirá o que fazer com os recursos. Já no contexto de commons este controle não se restringirá a uma única pessoa. Pelo contrário, em uma relação de acordo mútuo e coordenação baseada no mérito, todas as

pessoas que de alguma forma contribuem com o crescimento do commons tem direito a decisões sobre quais caminhos o mesmo irá tomar.

Há diferentes caracterizações de commons de acordo com sua abertura em relação a quem pode participar e com o nível de regulamentação interna aplicado. No entanto, o que define realmente o que é commons é o fato de que não há exclusões assimétricas de direito de uso. De fato, todos que participam do commons tem direitos iguais de uso sobre o mesmo, sem exclusões unilaterais. É a liberdade deste uso no âmbito individual, sem a necessidade de permissões por alguém, que define commons e é o motivo de seu sucesso.

Imre Simon e Miguel Said Vieira traduziram commons para o português como sendo rossio. Esse termo, que na definição do dicionário Houaiss (2001) tem o significado de um "terreno ou largo bastante espaçoso; grande praça" ou um "terreno roçado e usufruído em comum" (Simon, Vieira, 2008). Tanto a primeira quanto a segunda definição remetem ao uso que este nome teve em momentos históricos. Rossio, na antiga Lisboa, dava nome a uma famosa praça, hoje conhecida como Praça de D. Pedro IV, mas que no auge de sua efervescência foi ponto de encontro e discussão democrática entre os habitantes. Nos Estados Unidos este tipo de praça era conhecida como "towns commons", como o famoso Boston Common, criado originalmente como uma área de pasto mas que serviu durante muito tempo como ponto de encontro e de discussões políticas.

Na Inglaterra, durante a Idade Média, o termo commons deu nome à áreas de pasto usadas coletivamente em algumas comunidades e que tinham a interessante característica de não possuírem um proprietário exclusivo. O direito de uso e decisão sobre o futuro destas terras era compartilhado por todos que a usavam e se respaldava em parâmetros de sustentabilidade e transparência, buscando sempre o melhor para todos.

Esta revisão histórica serve para demonstrar que o conceito de commons não é algo novo ou exclusivo desta nova economia de informação conectada, mas sim uma idéia presente em diversos momentos da história. A diferença que esta nova economia dos commons apresenta é o seu alcançe global, garantido principalmente por esta grande rede que é a internet.

A produção social, ou peer production, aparece como uma faceta desta economia baseada em commons e se utiliza da mesma para emergir com a idéia de uma produção baseada em informação, descentralizada e auto-selecionada. Neste contexto, a ação de vários agentes distribuídos (pessoas), convergem e se auto-coordenam em um objetivo em comum, afim de produzir algo de interesse mútuo.

Nesta lógica cada indivíduo age como um agente produtor de acordo com seus próprios interesses. Ninguém produz algo, pelo simples fato de produzir, mas sim porque deseja algo em troca, seja dinheiro, reputação ou satisfação pessoal. A coordenação nasce do fato de que dado os interesses pessoais, estes são sinalizados ao grupo e democraticamente compreendem-se e

convergem a uma solução ótima.

Talvez o maior exemplo de produção social, hoje em dia, seja o conjunto de projetos que formam o open source. No capítulo IV e V, será explorada esta plataforma, tentando entender seus mecanismos afim de aplicar em outros domínios, como por exemplo a música. No entanto, há um exemplo de produção social baseada em commons atuando há bastante tempo, com vários agentes atuando em conjunto, de maneira descentralizada, sem organizações hierárquicas definindo o que deve ou não ser produzida, que é a própria ciência. O que esta nova econômia conectada fez foi intensificar dramaticamente este fenômeno em um número cada vez maior de plataformas e domínios de informação. (Benkler, 2006)

3. 3 MODULARIDADE E GRANULARIDADE

Segundo Benkler (2006, p.100), dois fatores influenciam diretamente no sucesso de uma iniciativa de produção social baseada em commons, como é o open source e como pretende ser o Hermeto. São elas modularidade e granularidade.

Modularidade se refere à quantidade de partes as quais um projeto pode ser quebrado e modificado sem que seja necessário a modificação em outras partes do mesmo. A esse segundo fator, dá-se o nome de acoplamento, termo bastante utilizado na engenharia de software. Essa característica é importante, porque segundo Benkler, permite que o número de pessoas trabalhando em cima de um mesmo projeto, ao mesmo tempo, cresça sem que o trabalho de um interfira no de outro.

A segunda característica, granularidade, indica o tamanho destas partes e está diretamente ligada ao tempo em que alguém precisará investir em uma única sessão de colaboração para contribuir com algo para o projeto. Isso garante uma maior adoção de colaboradores ao projeto já que o custo (horas) para participar é tão grande quanto for para contribuir com algo útil e completo. Este custo se torna claro em algumas iniciativas de produção social, como por exemplo, o tempo que alguém leva para contribuir na criação de um livro, como propôs a iniciativa dos Wikibooks, um site associado à Wikipedia e que incentivava a criação de livros de forma colaborativa. A iniciativa não deu certo e poucos textos chegaram à maturidade, muitos com a contribuição de apenas um autor. Isso claramente se distancia do sucesso obtido pela Wikipedia, aonde um artigo, muito menor e menos pretensioso do que um livro, é editado e moderado colaborativamente por muitas pessoas.

Como veremos estas duas características foram importantes na escolha da implementação e das tecnologias utilizadas no Hermeto (vide TiddlyWiki), como será visto no capítulo VII.

4 O OPEN SOURCE

Este capítulo tenta definir o open source buscando suas origens históricas e levantando os seus mecanismos técnicos internos. Serão levantadas algumas ferramentas básicas necessárias para o bom funcionamento e organização de um projeto distribuído como é o open source. O próximo capítulo dá uma ênfase maior aos mecanismos psicólogicos e econômicos do mesmo, tentando entender principalmente as motivações por trás do ato de contribuir.

4.1 OPEN SOURCE DEFINITION

Segundo a Open Source Initiative, um organização sem fins lucrativos que tem como meta revisar e aprovar licenças open source e que publicou a definição de facto para o open source - a Open Source Definition, o open source não significa apenas o acesso ao código fonte de um programa. Para um software ser considerado open source sua licença deve estar de acordo com dez critérios que dissertam desde a liberdade de distribuição, até a permissão para reutilizar e derivar este código em outros projetos.

"Open source is a set of principles and practices that promote access to the production and design process for various goods, products, resources and technical conclusions or advice. The term is most commonly applied to the source code of software that is made available to the general public with relaxed or non-existent intellectual property restrictions. This allows users to create user-generated software content through incremental individual effort or through collaboration." (Prieto)

Osterloh e Rota definem o open source nas seguintes palavras:

"*OS is a very successful innovation model that challenges conventional economic views that innovation is best supported by strong private intellectual property rights. (...) In OS, programmers place their software at the free disposal of other users and developers. The software is published on the internet and anybody interested can download it for free. Users are not only allowed to use the functionalities of the software, they are also permitted to view the source code of the programs. If they wish to change, debug or develop the code further, they are free to do so. They are even free to publish these amendments together with the original or the changed source code.* " (Osterloh e Rota, 2008)

Há muitas definições sobre o open source, vindas das mais diversas áreas. O que essas definições tem em comum é constatar a liberdade do acesso ao código fonte e a liberdade de se modificá-lo. Essas liberdades são protegidas por licenças que variam em maior ou menor grau a permissibilidade de tornar o comum em privado.

"*The most important differences refer to the extent to which they allow public property to be mixed with private property rights.*" (Osterloh e Rota, 2008)

Talvez a mais famosa delas seja a General Public License (GPL), criada e mantida pela Free Software Foundation (FSF). A GPL tem a interessante característica de "infectar" outros pedaços de software que utilizem um código-fonte protegido por seus termos com as mesmas liberdades que a define. Segundo esta licença, quem modificar ou reutilizar um código licenciado através da GPL terá que, obrigatoriamente, lançar o código resultante sob os mesmos termos dissertados pela GPL. Ou seja, terá que tornar público o código-fonte e garantir a liberdade para que outras pessoas possam reutilizá-lo ou modificá-lo, logicamente através da mesma licença. O resultado é uma espécie de cadeia virtuosa, em um fenômeno aonde surgem cada vez mais softwares e bibliotecas de alta qualidade e valor competitivo sendo publicados através desta licença e empresas privadas que teriam vantagem competitiva ao se utilizar destes códigos tendo que publicar seus produtos sob esta mesma licença.

4.2 ORIGENS

O software, por ser uma mídia relativamente nova, e por ter nascido e crescido em um ambiente propício ao compartilhamento, como as universidades conseguiu desenvolver uma história de apropriação e liberdade diferente das vividas por outras mídias como a música.

Suas origens coincidem com as origens dos hardwares e dos primeiros computadores, mas diferente destes, não teve seu potencial comercial explorado desde o início. As primeiras empresas que exploraram o mercado de computadores, viam no software apenas um utensílio para o funcionamento dos hardwares, focando nestes últimos seus reais modelos de negócio. O software era distribuído juntamente com o hardware e com eles muitas vezes o seu próprio código fonte.

Como muito dos clientes deste hardware eram também engenheiros e pesquisadores, a correção de bugs e modificações no código fonte eram enviadas de volta para as empresas produtoras que além de incentivarem esta prática, incentivavam a distribuição destes patches para outros usuários do mesmo hardware. (Fogel, 2009)

No entanto, apesar de haver menos restrições legais quanto ao compartilhamento, este prática era bastante dificultada por dois motivos (Fogel, 2009). A primeira era a falta de uma padronização na arquitetura dos hardwares da época, o que tornava a portabilidade das aplicações entre arquiteturas uma tarefa quase impossível. A segunda e mais importante é a própria inexistência de uma plataforma de compartilhamento como a possibilitada hoje pela internet.

Com o desenvolvimento da tecnologia de hardware, uma natural padronização começou a surgir, privilegiando aquelas arquiteturas que se estabeleceram como de facto, em parte por suas qualidades, em parte por seus investimentos em marketing.

Por outro lado, o surgimento das chamadas linguagens de alto nível e o desenvolvimento da tecologia de compiladores, a portabilidade entre arquiteturas se tornou ainda mais possível.

O resultado é que as empresas que produziam hardware começaram a enxergar no software uma oportunidade de agregar valor aos seus produtos através de funcionalidades inexistentes em outros hardwares. Com a comoditização do hardware o diferencial logo recaiu para o software vendido.

O compartilhamento se tornou então uma possibilidade perdida de ganhar vantagem entre concorrentes, seja porque a redistribuição de modificações entre usuários poderia retirar o valor agregado possibilitado pelo software, seja porque as modificações poderiam ser utilizadas pelos próprios concorrentes.

Para as empresas a alternativa natural foi fechar o código fonte, distribuindo apenas os binários, investindo cada vez mais em proteção legal através de direitos autorais.

4.3 GNU E A FREE SOFTWARE FOUNDATION

Ao mesmo tempo em que se iniciava a comercialização do software como um produto a parte surgiam também algumas iniciativas de resistência a esta prática. Há relatos de pelo menos duas correntes de programadores que mesmo com a crescente economia do software cultivaram a cultura do compartilhamento, vivenciada nos primeiros anos.

A primeira e mais politizada, no sentido de praticar o compartilhamento como uma forma de resistência ao fechamento dos códigos, foi a iniciada por Richard Stallman. Stallman vivenciou um momento de grande efervescência no desenvolvimento de software e na cultura de compartilhamento por ter trabalhado no laboratório de inteligência artificial no MIT durante os anos iniciais do desenvolvimento de software. O laboratório do MIT cultivava uma cultura de compartilhamento muito forte, como descreve o próprio Stallman:

"We did not call our software "free software", because that term did not yet exist; but that is what it was. Whenever people from another university or a company wanted to port and use a program, we gladly let them. If you saw someone using an unfamiliar and interesting program, you could always ask to see the source code, so that you could read it, change it, or cannibalize parts of it to make a new program." (Stallman)

Quando a onda da comercialização chegou, muitas empresas foram montadas e muitos colegas de Stallman no laboratório foram chamados para trabalhar nelas. Muitos levaram o próprio código fonte que era desenvolvido no laboratório, modificavam, faziam melhorias e vendiam como um produto, fechando o código fonte e não permitindo que ninguém o alterasse.

Stallman em resposta a essa tendência resolveu iniciar um projeto para implementar um sistema operacional totalmente livre dessas apropriações, iniciando a implementação de uma série de ferramentas necessárias para o funcionamento de um computador, dentre elas o compilador para a linguagem c, gcc, e o editor de textos Emacs. Fundou, então, a Free Software Foundation, uma organização sem fins lucrativos que tinha como objetivo agregar projetos de software livre.

Para evitar que a prática de apropriação pudesse ser realizada também publicou todas essas ferramentas através de uma licença que limitava esta possibilidade. A hoje conhecida General Public License (GPL), que dentre outras coisas permitia a total distribuição e modificação do código fonte de um programa desde que o mesmo fosse distribuido através da mesma licença.

Por outro lado, uma segunda iniciativa partiu de alguns programadores da Universidade de Berkeley na Califórnia, através do BSD (Berkeley Software Distribution), um sistema operacional baseado no Unix. Este grupo, com uma visão menos ideológica, não enxergava o compartilhamento como uma prática política, mas sim apenas uma maneira de produzir melhor código, com a ajuda de mais pessoas.

Essa diferença de pensamentos, entre motivações ideológicas e meramente práticas, irão resultar em uma espécie de segmentação no mundo do software aberto. Enquanto que a prática do compartilhamento como uma prática política, chamada software livre ou free software, irá instigar e motivar muitos projetos de software, por outro lado, muitos outros programadores enxergarão esta prática apenas como uma maneira de otimizar suas buscas por soluções para seus problemas, através do crescente número de projetos que já se encontram com uma base de código madura, com uma comunidade atuante na correção de bugs, e que irá servir a seu propósito, evitando ter que "reinventar a roda". Eventualmente a satisfação encontrada através desses projetos irão motivá-los a contribuir para o mesmo como uma maneira de retribuir o que receberam inicialmente. Esta segunda tendência, conhecida apenas como "open-source", faz o uso de licenças menos restritivas quanto a necessidade de ter que publicar o software que usou o projeto com a mesma licença do projeto. Isso permite, por exemplo, a utilização desses softwares em uso comercial, em empresas que não querem abrir o código fonte de seus produtos.

4.4 ELEMENTOS DO OPEN SOURCE

Fogel, em seu livro "Producing Open Source Software: How to Run a Successful Free Software Project", declara que:

"Free software projects rely on technologies that support the selective capture and integration of information. The more skilled you are at using these technologies, and at persuading others to use them, the more successful your project will be." (Fogel)

Ou seja, para que um projeto open source alcançe o sucesso desejado, ele precisa de tecnologias que permitam a comunicação e integração de informações entre seus participantes. Por ser uma atividade totalmente distribuída, onde as pessoas, informações e infra-estrutura, estão em diferentes lugares ao redor do globo, é extremamente necessário meios que permitam a coordenação de tarefas remotamente.

Ao longo de sua história, o open source convergiu para uma série de padrões e ferramentas que facilitaram esta integração. Nos próximos tópicos serão descritas algumas delas, bem como alguns padrões de uso. O objetivo é tentar levantar as ferramentas necessárias para que a colaboração em outros domínios, como a música, possa ser facilitada.

No capítulo em que será descrito a implementação do Hermeto, também serão discutidos o uso de algumas dessas ferramentas na implementação do mesmo.

4.4.1 O que um projeto precisa?

Fogel define um conjunto básico de ferramentas que a maioria dos projetos open source utilizam. São elas: um web site, uma wiki de referência, uma lista de emails, um sistema de controle de versão, um bug tracker e um canal de comunicação em tempo real, normalmente realizado através de chats ou mensagens instantâneas. Nem todos os projetos se utilizam de todas essas ferramentas, mas a maioria dos projetos de sucesso possuem todas essas ferramentas.

a) Web Site

A maioria dos projetos precisa de uma referência na web aonde novos colaboradores ou usuários possam encontrar informações sobre o projeto. Um web site é a maneira mais comum de completar este requisito, já que é possível encontrá-lo através de sites de busca e facilita a propaganda do projeto através da divulgação de um endereço único. Entre as ferramentas citadas é a mais básica e talvez a mais importante, pois tem a capacidade de centralizar as informações necessárias para o uso do projeto. Normalmente é o local aonde é hospedada a documentação do projeto, bem como links para repositório de código, endereço da lista de emails, endereço do bug tracking, além de um resumo das propostas do projeto. Em alguns casos é substituída por uma wiki.

b) Wikis

Wikis são ambientes de colaboração de conteúdo extensível que trazem consigo o revolucionário conceito de que qualquer pessoa no grupo de colaboradores pode criar e editar novos conteúdos a qualquer momento. É bastante utilizado em projetos open source como um meio de manter as documentações do projeto em um local acessível e atualizado com frequência. Será também a base do sistema implementado neste trabalho.

c) Lista de emails

Uma lista de email é utilizada como o principal meio de comunicação em um projeto open source. É através da lista que são debatidas questões sobre o encaminhamento do projeto, decisões de design, visões sobre o futuro e o presente do mesmo. Além disso é o meio mais rápido de um usuário tirar dúvidas sobre questões técnicas. Normalmente um projeto com uma grande base de usuários possui mais de uma lista de emails, uma para os desenvolvedores, aonde são debatidas questões sobre o desenvolvimento do projeto e uma para usuários, aonde usuários e desenvolvedores se misturam em um único canal de comunicação para esclarecer e responder dúvidas.

d) Sistema de controle de versão

Não há desenvolvimento de software profissional hoje em dia que não faça uso de algum sistema de controle de versão. É impraticável pensar em desenvolvimento de software colaborativo, hoje em dia, sem um controle pragmático de versões, seja no mundo open source, seja no mundo comercial de software fechado.

O controle de versão nada mais é do que um gerenciamento das versões de um arquivo de código fonte. Isso significa gerenciar as modificações feitas em cima do arquivo ao longo de sua existência. O sistema de controle de versão consegue automatizar este procedimento permitindo, entre outras funcionalidades, mesclar versões diferentes de um arquivo (o chamado merge), criar patches (conjunto de modificações feitas em mais de um arquivo, muito utilizado no open source para revisar contribuições feitas por contribuintes), sincronizar o uso concorrente entre mais de um usuário, gerenciando permissões de acesso e modificação, além de outras tarefas.

Fogel (1999) aconselha o versionamento não só de código fonte, mas de tudo que envolve edição colaborativa em um projeto. Desde documentação, FAQs, web sites, até notas de design. Ele aconselha também que tudo que esteja versionado fiquem próximos, no mesmo repositório. A razão disso é que dessa maneira irão evoluir juntamente com o código fonte. Quando for necessário voltar para alguma versão mais antiga do software (para resolver bugs ou por outros motivos) a documentação será referente a mesma versão do código fonte.

Conteúdos que não mudam, como emails, devem ser arquivadas (tarefa realizada pela maioria das ferramentas de listas de emails). Arquivos gerados automaticamentes não devem ser versionados (como por exemplo, arquivos de configuração gerada por sistemas de build).

Ainda em Fogel, o mesmo acha de extrema importância o acesso fácil e navegável do repositório através da web. O motivo é a facilidade de inspeção do código sem a necessidade de ter que instalar um sistema de controle de versão na máquina para baixar o repositório e somente assim ter acesso ao código. A importância disso se reflete principalmente na comunicação entre o time de desenvolvedores e até mesmo usuários nas listas de emails. Segundo ele, é muito mais fácil alguém apontar alguma modificação em um arquivo ou algum bug encontrado quando este arquivo está acessível através de uma url.

- Branches

Outra funcionalidade disponibilizada por sistemas de controle de versão é a possibilidade de desenvolvedores realizarem branches. Branches são ramificações feitas em cima da base de código principal. Estas ramificações são de extrema importância pois permitem que desenvolvedores experimentem novas funcionalidades no projeto, sem interferir no ramo principal de trabalho (conhecimento como trunk). Caso a experimentação alcançe resultados positivos ela então é integrada novamente no trunk, em uma operação conhecida como merge (mescla). Qualquer sistema de controle de versão moderno possibilita todas essas operações de maneira semi-transparente para o desenvolvedor.

O branch tem mais uma utilidade que é a de marcar versões do software consideradas como estáveis. Essas marcações, conhecidas como tags em alguns desses sistemas, facilita o arquivamento de versões ao longo do tempo. Se alguém quiser voltar para alguma versão, para verificar a introdução de algum bug, por exemplo, vai ter através dessa funcionalidade uma

valiosa ajuda.

e) Bug Tracking

Tracking é o ato de controlar e gerenciar tarefas em um projeto. No mundo do desenvolvimento de software é muito utilizado como um ponto de entrada para requisições de novas funcionalidades, bem como relato de bugs (defeitos). Após priorizada pelo time de desenvolvedores, um ticket, como é chamada uma entrada no bug tracking, é associado a um ou mais desenvolvedores. Após qualquer novidade no status da tarefa os desenvolvedores atualizam o ticket com o novo estado. Dessa maneira, interessados na resolução do bug ou na finalização da funcionalidade ficam a par do que está acontecendo.

Há alguns padrões de uso estabelecidos para o bom funcionamento dessas ferramentas, normalmente referentes à especificação da solicitação feita. No caso de um bug, isso representa conseguir reproduzir a situação que demonstra o bug através das informações disponibilizadas no ticket. No caso de solicitação de novas funcionalidades refere-se à especificação do que se deseja conseguir através da mesma. Com exceção de algumas pequenas variações, o ciclo de vida de uma solicitação em um bug tracker segue a seguinte ordem:

1. Alguém entra com um novo ticket. Este alguém pode ser um usuário ou um desenvolvedor. As primeiras informações são preenchidas, como um resumo da situação, a descrição do que se deseja e os passos para conseguir reproduzir, caso seja um bug. Neste momento o ticket é considerado em um estado não verificado.

2. Outras pessoas analisam e adicionam comentários ao ticket.

3. Outra pessoa reproduz o bug. Este momento é importante pois demonstra que o relato é verdadeiro. Normalemente este item e o próximo estão intimamente ligados.

4. Dependendo do projeto há desenvolvedores que priorizam o ticket, classificando sua importância e urgência para o funcionamento do software. Normalmente, esses desenvolvedores fazem parte do time principal de desenvolvimento.

5. Eventualmente alguém diagnostica o problema e passa a ser o dono do ticket, ou seja, seta o seu nome como alguém que está trabalhando no problema.

6. Após finalizado, o dono do ticket faz o upload de suas alterações para o repositório de controle de versões (ato conhecido como commit) e seta o ticket como resolvido. O sistema de tickets então notifica aqueles que se cadastraram como interessados na resolução do ticket.

Como foi dito, algumas variações acontecem de projeto para projeto. O importante é que se encontre uma forma de trabalhar que satisfaça os usuários e os desenvolvedores.

g) Chat

Alguns projetos se utilizam de chats, ou sistemas de mensagem instantânea, para facilitar a comunicação em tempo real entre os integrantes e usuários do projeto. A ferramenta mais utilizada para esta tarefa ainda é o conhecido IRC (Internet Relay Chat), pela rapidez na comunicação e por permitir a criação de salas de chat aonde todos se comunicam com todos.

5 MOTIVAÇÕES NO ATO DE CONTRIBUIR

Mas o que está por trás dos projetos de software aberto? O que leva uma pessoa a participar na criação de um pedaço de software o qual não a pertence (ou não a pertence exclusivamente), muitas vezes sem ser pago para isso? Quais os benefícios - materiais ou não - que ela pode receber contribuindo? E que mecanismos estão por trás destas comunidades? Como funcionam internamente? Quais são os seus meios de coerção e gerenciamento? E mais importante, o que se pode tirar como experiência destas comunidades para o uso em outras atividades criativas? É possível alguma generalização dos seus mecanismos para que possam ser reutilizados no desenvolvimento do Hermeto? É a essas perguntas que este capítulo pretende responder.

5.1 CARACTERÍSTICAS ECONÔMICAS DO OPEN SOURCE

Uma das características do sucesso do open source é o fato do código fonte ser também um bem não-rival. Por ser um objeto de informação, depois de ter sido publicado através de uma licença open source, o seu uso por alguém não exclui de maneira alguma o uso por outrem.

Outra característica importante é o modelo de inovação proposto pelos projetos open source. Von Hippel and von Krogh (2003). definem este modelo como um modelo "coletivo-privado". Ou seja, os participantes deste modelo investem seus recursos privados (principalmente tempo), para a produção de um bem coletivo. Este modelo se difere dos dois principais modelos tradicionais de inovação, o completamente privado e o completamente público ou estatal.

No investimento privado os participantes produzem inovação apenas se os benefícios econômicos puderem pagar o investimento. Para isso, seus participantes irão tentar evitar ao máximo a disseminação do conhecimento produzido para o público, através de patentes e leis de proteção ao direito autoral.

Já no investimento público o principal investidor é o estado, que através de incentivos financeiros ou premiações na forma de reputação, permitem a produção de pesquisas e inovações que serão de interesse para a sociedade. É interessante notar que neste modelo os participantes do processo são alguns poucos escolhidos, pesquisadores de universidades ou fundações, que recebem o direito de produzir inovação.

"Careful selection of contributors is important here (n.a: no modelo de investimento público), so that only individuals who are sensitive to these selective incentives are chosen." (Osterloh e Rota, 2008)

Já no modelo "coletivo-privado" do open source esta seletividade é praticamente nula, sendo tarefa do próprio participante (que na prática pode ser qualquer pessoa), verificar se a sua participação trará benefícios para o coletivo e para si mesmo.

Segundo Von Hippel and von Krogh (2003) o modelo "coletivo-privado" está em um meio termo entre os dois modelos tradicionais. Os participantes revelam suas inovações sem exigirem propriedade intelectual, mas não há um agente central, como o estado, selecionando quem deve receber os recursos para participar. Além disso, a participação neste modelo deve ser recompensável para o próprio participante. O ato de contribuir é realizado somente se for de interesse para o contribuinte, seja este interesse econômico, social ou apenas pessoal.

"Contribution has to be self-rewarding, i.e. only programmers will contribute whose utility is greater when contributing than when free-riding. In that way, OS seems to unite the benefits of the private investment and the collective action model while avoiding their downsides." (Osterloh e Rota, 2008)

5.1.1 Cultura da dádiva

Para entender melhor as motivações por trás do ato de contribuir será preciso antes uma breve explanação sobre o que é a cultura e economia da dádiva, nas quais o open source está inserido. Eric Raymond (2001), em seu livro "The Cathedral & The Bazaar", descreve o ser humano como um inato competidor por status social. Segundo ele, esse desejo se dá por uma própria necessidade evolucionária, onde o ser mais bem visto pelos outros teve sempre acesso aos melhores recursos, como comida e remédios.

Essa característica do ser humano irá guiá-lo nas mais diversas formas de organização que aparecem durante a sua história, variando em intensidade de acordo com o nível de escassez desses recursos.

"This drive for status express itself in different ways, depending largely on the degree of scarcity of survival goods." (Raymond, 2001)

Raymond continua, descrevendo três das principais formas de organização encontradas ao longo da história. Na primeira e mais primitiva, a "hierarquia de comando", os bens e recursos mais escassos são organizados e distribuidos por uma autoridade central que impõe-se muitas vezes através da força. O status social é reconhecido através dessa coerção de poder.

Grande parte do que conhecemos hoje como economia de mercado e neoliberalismo podem ser sintetizados por uma outra forma de organização, a "economia de troca". Nesta segunda forma, a alocação de bens e recursos escassos é realizada principalmente através da troca em uma hierarquia descentralizada. O status social surge como uma extensão dessa necessidade de troca e é determinada por aqueles que conseguem o maior número de bens, sejam materiais ou não.

É interessante notar que apesar de hoje vivermos em uma economia de troca, ainda assim é possível encontrar formas de organização próximas de uma hierarquia de comando, como o próprio governo, militares e até mesmo o crime organizado.

Uma terceira forma de organização, no entanto, surge em populações que ao invés de apresentarem escassez, apresentam abundância de seus recursos de sobrevivência. Esse fenômeno pode ser encontrado em algumas populações de índios, que pouco sofreram com a invasão do homem, em algumas sociedades filantrópicas de milionários ou nas comunidades de projetos open source.

Nessas populações o status social não é determinado pelo controle dos bens e recursos, mas sim pela quantidade de bens e recursos que se contribui de volta para a comunidade. Essa terceira forma é a chamada "cultura da dádiva".

"In gift cultures, social status is determined not by what you control but by what you give away." (Raymond, 2001)

Não há dúvidas que projetos open source se enquadram nesta cultura. Com a ubiquidade das redes e dos meios tecnológicos, é possível afirmar que recursos como equipamentos e acesso à informação (matéria-prima desses projetos) realmente são pouco escassos. No entanto, obviamente as pessoas que contribuem em um projeto open source investem tempo (e consequentemente dinheiro) em cima de sua participação. O simples fato de pertencer a uma economia de dádiva não explica a motivação por trás do ato de contribuir. Como veremos qualquer cultura de dádiva que deseje ser economicamente viável precisa ter uma motivação a mais do que puramente status social.

5.1.2 O produtor é o usuário

No mesmo livro descrito acima, Eric Raymond inicia seu ensaio sobre o open source, definindo a principal motivação, pela qual segundo ele, alguém inicia um projeto de software:

"Every good work of software starts by scratching a developer's personal itch." (Raymond, 2001)

Não é que todo o pedaço de software lançado como open source seja uma real necessidade para o desenvolvedor, mas com certeza ele tem interesse em ver o problema que o software se propõe a solucionar, resolvido. Seja por motivos pessoais diretos, seja porque ele está sendo pago para isso. O fato é que como produtor de software, ao contribuir em um pedaço do mesmo ele, está na verdade contribuindo para uma necessidade sua.

Ou seja, outra característica fundamental para o sucesso do open source está no fato de que o produtor do software, o desenvolvedor, é ele mesmo o próprio usuário. Quando um participante de um projeto open source decide corrigir um bug ou desenvolver uma nova feature, não é o sentimento de filantropia ou o desejo de fazer um mundo melhor que aparece como principal motivador. Pode haver sim esta necessidade, assim como há o fator satisfação pessoal, como será discutido mais a frente. No entanto, acredita-se que um dos principais motivos para o sucesso do open source como plataforma de inovação e criação é o fato de que os produtores (desenvolvedores, tradutores, etc) são os próprios usuários. E isso faz toda a diferença.

"Users can be a rich source of (especially incremental) innovation (von Hippel, 1988). By working with products they get to know them intimately, detect their shortcomings and develop ideas how the products could be altered to better suit their needs." (Osterloh e Rota, 2008)

Neste ponto, vale analisar dois tipos de contribuição que podem ser feitas para a comunidade. A primeira é a contribuição ocasional. Neste caso uma pessoa, que não está entre aqueles que iniciaram o projeto, descobre no mesmo uma oportunidade para ajudá-lo em outro trabalho que esteja fazendo, reduzindo, no uso do projeto open source, a necessidade de ter que reimplementar do zero uma funcionalidade ou ferramenta que a comunidade já contribuiu para ele de graça. Neste caso é natural que qualquer melhoria que esta pessoa faça no código seja contribuída de volta para a comunidade, já que ele está totalmente satisfeito de ter economizado uma boa parte do trabalho que ele precisaria fazer.

A segunda forma de contribuição é aquela feita pelos próprios donos do projeto. Esta situação não é tão natural como pode se pensar no caso de uma contribuição ocasional, já que ao doar à comunidade o código fonte e todo o trabalho envolvido na produção do mesmo, os donos (muitas vezes com empresas por trás) estarão abrindo mão de certas vantagens competitivas caso este código fosse mantido fechado.

No entanto, em uma análise mais profunda pode se encontrar vantagens que superam aquelas conseguidas com o fechamento de um código e sua venda como produto fechado. Essas vantagens ultrapassam o simples fator econômico, adentrando o escopo do desenvolvimento pleno do ser humano e evolução da humanidade.

5.1.3 Sobre os ombros de gigantes (On the shoulders of giants)

A produção de informação e conhecimento (como é o caso do software e também da música) se difere da produção de bens tangíveis no sentido em que se fundamentam em dois princípios. O primeiro é o próprio fato de serem bens intangíveis e consequentemente não-rivais. O segundo é o chamado efeito dos "ombros dos gigantes".

O termo é tirado da célebre frase de Isaac Newton quando se referiu ao seu sucesso como pesquisador ao fato de não ter feito tudo sozinho, mas sim que só se tornaram possíveis suas descobertas porque estava apoiado sobre ombros de gigantes. Newton se referia a todos os grandes pensadores que vieram antes dele, mas o que nos interessa aqui é a generalização que isto traz sobre este tipo de bem intangível e não rival - o de que há um custo intrínseco a este tipo de produção que é o que Benkler definiu como "input information", qual seja, todas as informações, conhecimentos e influências necessárias para que o criador de hoje possa produzir coisas novas.

No mercado de software esse custo se torna bastante visível. Qual seria o custo real de treinamento pessoal se não existisse a quantidade de informação técnica disponível e de graça na internet (interrogação) Ou se uma empresa de roteadores precisasse implementar todo um novo sistema operacional se não existisse o FreeBSD (interrogação)

Isso traz uma conclusão econômica sobre a produção de conhecimento. Quanto mais fechada forem as políticas de propriedade intelectual mais interessante se torna para as empresas também se fecharem. Mais elas irão fundamentar seus modelos de negócio à exploração do direito ao uso através do direito de autor. Isso leva a uma ineficiência econômica muito grande e um prejuízo enorme para a humanidade em geral ao restringir a quantidade de informação disponível para a evolução e disseminação do conhecimento.

5.1.4 Information inputs

Segundo Benkler (Benkler, 2006, p.52), a produção de informação e cultura tem três grandes "information inputs". O primeiro é a informação pré-existente e disponível como subsídio para a criação de novas informações e conhecimento. O segundo é a tecnologia envolvida e necessária para a produção, quais sejam, meios de transmissão, meios de gravação, instrumentos e ferramentas. O terceiro é a própria capacidade humana. Isso envolve a criatividade e outras qualidades do ser criativo. Sendo que a informação é um bem não-rival, e tem seu custo aproximado a zero quando ela não é proprietária, e a ubiquidade das redes e o barateamento do poder de processamento tornaram o segundo input extremamente barato o que sobra como grande diferencial é a própria capacidade humana de transformar esses inputs em conhecimento,

insights e ciência.

E essa capacidade humana é encontrada com muito mais facilidade e intensidade em um nível global como a internet possibilita do que os recursos humanos de um empresa fechada nunca sonharia obter.

5.1.5 Rivalização do autor

É nesse ponto que o open source parece se distanciar de outras plataformas tradicionais de criação como a indústria da arte.

A arte e específicamente a música, também são produtos do conhecimento humano, mas não o simples fruto de um ou outro indivíduo específico. É sim o fruto de uma sucessão de contribuições feitas desde o primeiro homem, expressando a história de sua caça em uma caverna, até os dias atuais.

"The act of artistic production doesn't come from nowhere; neither is it born in the heads of private individuals. It doesn't dwell in a social nothingness. Nor does it start with a blank canvas. Any moment of production involves the reassembling and rearranging of the diverse materials, practices and influences that came before it and which surround it. "(Berry e Moss, 2008)

O indivíduo criativo, assim como o colaborador de um projeto open source, tem seu mérito em conseguir transformar os conhecimentos e sentimentos que estão disponíveis a ele na forma da consciência coletiva em um produto final digerível e belo.

No entanto, o que deveria ser uma atividade natural do ser humano, uma extensão à sua existência, que a realiza e lhe dá significado, acabou nos últimos séculos sendo restringida a poucos privilegiados que ganharam o direito exclusivo de criar e produzir sob a batuta dessas indústrias de criatividade.

Esse fenômeno, o qual define-se aqui como rivalização do autor, se explica pela necessidade, por parte dessas indústrias, de tornar uma característica intrínseca à existência humana, em algo escasso e raro, na conhecida lógica da oferta e procura.

Para que seja comercialmente viável a atividade criativa precisa, segundo esta lógica, ser comercializada como propriedade. É aí que entram as leis de direitos autorais, protegendo a criação como propriedade, a princípio do autor, mas que logo é transformada em propriedade de quem financiou o autor, ou seja seus patrocinadores, as indústrias.

Berry e Moss resumem isto neste parágrafo de seu artigo "Art, Creativity, Intellectual Property

and The Commons":

"(...) in order to generate profit from art, creative products must be transformed into property that may be owned and exchanged by private profiteers. “Intellectual property rights,” enforced by the state, are the mechanism for achieving this. Intellectual property requires a legal persona who “owns” the creative product in order to function: the “author.” This legal fiction is the sovereign “individual,” endowed with the power of creation, someone who “justifiably” has ownership rights over their creative goods and “deserves” to be handsomely rewarded. These creative goods, even though they were created in and out of the public, may then be owned by private entities (and not necessarily the “original creators”) and removed from what we share in common. Then, as property, these creative products can be exchanged among private hands, and traded and consumed in the market place." (Berry e Moss, 2008)

5.1.6 Work versus Labour

Nos tópicos anteriores foram enumeradas algumas teorias para o ato de contribuir para um bem comum como o open source. Essas teorias variam desde motivações econômicas, como a existência de um custo intrínseco à produção de conhecimento, os "informations inputs", até motivações mais práticas, como a característica de que no open source o produtor é o próprio usuário.

No entanto, há uma explicação defendida por alguns pesquisadores, fundamentada em teorias do trabalho humano e na importância do mesmo para a realização existencial das pessoas. Estas teorias são importantes para esta pesquisa ao aproximar claramente a atividade criativa vivenciada pelo open source da atividade artística, aqui representada pela música.

Segundo esta linha de pesquisa há uma diferença importante entre duas formas clássicas de enxergar o trabalho humano. O trabalho que realiza e dá significado à existência, aqui denominado emprego e o trabalho puramente mecânico e superficial, o qual chamaremos de labuta.

Essa distinção não é nova e já foi vivenciada e discutida em várias épocas. Os gregos na antiguidade clássica, por exemplo, distinguiam o trabalho dos escravos do trabalho dos artesãos. Hannah Arendt em seu livro The Human Condition (1998) define a labuta como uma atividade que não apresenta evolução humana a quem o pratica. São trabalhos que não produzem nada além do necessário para a sobrevivência. Não deixam frutos e nem produzem coisas belas.

Marx apresenta essas distinção em vários momentos de sua obra. Sua teoria da alienação no trabalho apresenta em todos os momentos exemplos desse fenômeno em um contexto de

produção industrial. Sem enxergar o produto final de seu trabalho em linhas de produção completamente segmentadas os trabalhadores industriais acabam não se realizando como seres humanos completos, alienados sobre os produtos de suas próprias mãos.

De fato, a sociedade atual é uma sociedade fundamentada no trabalho laboral. Grande parte do que fazemos está intimamente ligada ao fato trágico da sobrevivência, onde trabalhos cada vez mais maçantes e distantes da atividade criativa surgem como necessidade de um estilo de vida baseado no conforto e segurança. Toda a atividade que não se encaixa nessa linha é considerada menor ou hobby e delegada para os mínimos momentos de lazer.

"(...)whatever we do, we do for the sake of “making a living". Anything that cannot be classified as part of making a living becomes a “hobby,” subsumed under playfulness or non-serious activities. (Berry, 2008)

Por outro lado, aquele que produz e vê o fruto de seu trabalho com orgulho, este é capaz de se realizar completamente, pois o trabalho é a extensão da existência, a motiva e a significa, deixa rastros de si para as gerações futuras.

É nesse ponto que o desenvolvimento de software se aproxima da atividade artística. A arte é o exemplo máximo de trabalho que deixa rastros. Criar arte é procurar o belo na existência. É a busca incessante pela repetição do momento sublime da criação inicial, quando o homem explica o motivo de sua existência, dando a ela um significado especial para si e para os que o rodeiam.

De certa forma o desenvolvimento de software tem muitas semelhanças com a arte. Alguns, como Donald Knuth em The Art of Programming (Knuth) chegam a chamá-lo de atividade artística. O fato é que ambos são trabalhos que envolvem criação e criatividade, além de senso estético e bom gosto. Ambos exigem concisão e senso crítico e, principalmente, é possível enxergar claramente o produto final como fruto do trabalho de quem o realiza.

"The care that goes into much free software production contributes to its craftsmanship — the production of code is in many ways “public,” and the source code can be read and admired by others, in this respect it seems to be similar to a great speech or a work of art." (Berry, 2008)

Daí vem uma das explicações para o fenômeno da contribuição no desenvolvimento de software aberto. Assim como o artista ou artesão que cria sua obra mais como meio de realização existencial como ser humano e menos como necessidade financeira, assim também o é o desenvolvedor que gasta seu tempo livre contribuindo em um projeto open source. Ele faz porque espera em troca a realização de participar da criação de algo que considera belo e que deixará

como legado para que outras pessoas usem e apreciem.

5.1. 7 Satisfação pessoal

Tudo isso nos leva a uma conclusão fundamental sobre a viabilidade (no sentido de ser possível existir), de um fenêmeno como o open source. Ela só é possível porque todos os objetivos listados acima envolvem e podem ser resumidos pela pura e simples satisfação pessoal. Toda e qualquer motivação secundária envolvida na participação do open source são apenas meios para o fim maior que é a realização dos envolvidos como seres humanos.

6 COLABORAÇÃO EM ARTE

Os capítulo I e II demonstraram que novas possibilidades de criação colaborativa estão surgindo, em parte devido às inovações das tecnologias de redes e o barateamento dos dispositivos necessários para participar das mesmas. O desenvolvimento de software foi a primeira atividade a se beneficiar destas possibilidades, talvez por seus participantes serem os próprios criadores destas tecnologias. Mas será que é possível levar estas idéias para outros campos criativos (interr.)

O que o presente trabalho pretende provar é que é possível sim, e em especial a arte e música, por serem atividades essencialmente sociais, no sentido de que seus participantes precisam interagir entre si e com o público para que o momento artístico aconteça. Cada vez mais a idéia da arte nascida da batalha individual do artista com a sociedade, se torna obsoleta, em troca de uma criação nascida da coletividade e da interação entre as pessoas.

"É a hora da poesia sem morte como consequência (Gingsberg, 1969)"

Como os próximos tópicos irão demonstrar, já aconteceram em vários momentos na história da arte, movimentos que tentaram aproximar a arte de uma idéia de criação coletiva e colaborativa, com resultados bem satisfatórios. Em comum, todos esses movimentos tem a crítica implícita à esta tradição de criação individual. Parafraseando o site collabarts.org, um site que abriga artigos sobre o tema:

"Comum a maioria se não a todas práticas colaborativas é a crítica implícita à idéia do artista como uma figura que se abriga às margens da sociedade engajado em um interno e solitário diálogo".

6.1 ARTE POSTAL

Uma das primeiras iniciativas em colaboração em arte através de redes sociais foi o movimento de artistas postais, com origens na década de 60. O movimento em si consistia em uma rede de artistas, (em sua maioria artistas plásticos e escritores) que se comunicavam através de correspondências postais e que faziam das próprias cartas enviadas o objeto de sua arte. Seus participantes enviavam entre si desde contos literários, gravuras, pinturas até objetos mais elaborados como pequenas esculturas. O próprio envelope era utilizado como parte integrante da obra.

Apesar de experiências similares terem acontecido no trabalho de artistas como Pablo Picasso, Henri Matisse, Marcel Duchamp, Francis Picabia (Uma introdução à Arte Postal), foi no ano de 1962 com o trabalho de Ray Johnson e sua "New York Correspondance School of Art", uma instituição criada com o intuito de formalizar o uso dessa mídia (a carta) como forma artística.

O interessante aqui é o pioneirismo em relação ao conceito de rede social. Muito antes de empresas de tecnologia se beneficiarem deste conceito para criar produtos altamente rentáveis, estes grupos já se organizavam através de redes de correspondência em busca de uma alternativa para mostrar e compartilhar suas produções criativas e mais importante, colaborar entre si na criação de algo coletivo.

A colaboração sempre esteve presente nas experiências de arte postal. Trabalhos que precisavam ser continuados por quem os recebesse eram uma prática comum nestes meios. Listas de endereço eram criadas para manter um local onde novas pessoas poderiam se inscrever para participar.

"Todos estes elementos (na: os elementos da arte postal), (...), vão produzir trabalhos onde as criações não privilegiam mais as obras como mercadoria, mas como um produto de comunicação onde se destacam os elementos da interatividade, da produção coletiva que gera a possibilidade de co-autoria, da obra em constante modificação, portanto, em processo, diante de redes que privilegiam o intercâmbio político e cultural. (Uma introdução à Arte Postal)

6.2 COMPOSITORES DE MÚSICA

Outra grande experiência em colaboração em arte surgiu na própria música através dos compositores, principalmente os compositores populares. Em detrimento à uma tradição de compositores clássicos solitários (ver Bethoven, Mahler, etc), a música popular sempre foi mais aberta ao aparecimento das chamadas parcerias.

Exemplos não faltam. Em uma busca rápida logo surgem nomes como Lennon / McCartney, Ira / George Gershwin, Tom Jobim / Vinícius de Morais, entre outros.

A própria idéia de conjunto musical, tão disseminada na música popular, demonstra uma espécie de criação colaborativa ao permitir que o grupo esteja o tempo todo realizando contribuições entre si com o objetivo de produzir algo, no caso a música.

Colaboração à distância com a ajuda de tecnologias (mesmo que ainda precárias), também já é algo bem experimentado por alguns compositores. É notável, por exemplo, as parcerias realizadas através de gravações de fitas ou até mesmo pela linha telefônica, sem contar a própria

comunicação por cartas.

Na prática, o que se percebe é que a maioria das artes são na verdade atividades coletivas. É o caso do teatro, da música e do cinema. Outras artes como a literatura ou as artes plásticas são atividades tradicionalmente mais individuais, talvez em parte pela própria dinâmica das mesmas. No entanto, como mostrado neste capítulo há exemplos notáveis do emprego da colaboração nestas artes em favor de uma prática criativa mais social e próxima da realidade.

No próximo capítulo, quando dissertado sobre o Hermeto, serão recapitulados alguns sistemas existentes que tentam facilitar estas parcerias no âmbito da música mas já com a ajuda de tecnologias mais recentes.

7 DESIGN E IMPLEMENTAÇÃO DO HERMETO

Depois de dissertado sobre o contexto da economia de informação interconectada, suas implicações no processo criativo, mais especificamente no processo criativo do open source e sua relação com a música, é hora de algo prático.

Como prova de que é possível uma criação musical colaborativa e distribuída nos moldes do open source, propõe-se aqui o desenvolvimento de um sistema colaborativo de produção musical, o Hermeto.

Vale lembrar que o sistema proposto é apenas uma prova de conceito, de maneira alguma tendo o objetivo de ser um sistema completo e robusto, mas sim o de tentar trazer algumas idéias de produção colaborativa, inspiradas pelo open source, para um ambiente musical. Até mesmo funcionalidades consideradas básicas para o uso real de um sistema colaborativo ficarão fora do escopo deste trabalho, podendo posteriormente surgir como trabalhos futuros.

7.1 TRABALHOS EXISTENTES

Até o momento da escrita deste trabalho algumas iniciativas já haviam se utilizado da tecnologia para abordar este tema de colaboração em música. Algumas são soluções pagas, outras de graça, algumas rodando no browser, outras exigindo a utilização de um aplicativo desktop. O que todas tem em comum é o fato de se utilizarem da internet como plataforma de colaboração. A seguir serão listadas algumas delas.

- Res Rocket

A primeira grande experiência conhecida de colaboração em música com o auxílio da internet foi o sistema Res Rocket. Sua primeira versão, em 1994, refletia as limitações tecnológicas da época, mas se aproveitou de maneira pioneira dos recursos disponíveis. Baseado na troca de arquivos no formato MIDI, algumas pessoas começaram a compartilhar idéias musicais através de uma lista de emails e um servidor FTP. Após algum sucesso, alguns integrantes implementaram um primeiro cliente aplicativo para o sistema. Com o nome de Res Rocket Surfer, o sistema consistia principalmente em um sequenciador MIDI, aonde organizados em salas distintas, um misto de salas de bate-papo e salas de estúdios, os participantes se comunicavam e compartilhavam trechos de suas composições. (A Produção Musical e os paradigmas colaborativos na Sociedade do Conhecimento)

Em 1999, com o grande sucesso e um crescente número de usuários, foi lançado um single com composições feitas no ambiente. O single, denominado War Child era adquirido através de uma doação de US$1,99.

Após a aquisição do mesmo por empresas como a Steinberg e Avid/Digidesign e devido a uma falta de visão quanto as potencialidades de um sistema como esse, o Res Rocket praticamente encerra suas atividades, sobrando apenas alguns grupos de usuários, organizados em listas de emails.

- Indaba Music

Construída em cima da idéia de redes sociais, o site Indaba Music permite o compartilhamento e edição de músicas no formato mp3, em um ambiente web, aonde seus usuários podem criar perfis próprios, buscar usuários com perfis semelhantes e interagir com outros membros. O site oferece o serviço de graça para usuários básicos e cobra uma taxa para usuários que desejam opções mais avançadas.

- Ejamming Audio

eJamming Audio é a combinação de um site de relacionamentos com um aplicativo desktop que permite aos seus usuários plugarem seus instrumentos e participarem de "jams sessions" em tempo real pela internet. Membros podem se classificar entre si, de acordo com o nível técnico. É possível gravar as sessões mas até o momento não era possível fazer o upload de músicas. O site é pago.

7.1.1 Diferenças entre o Hermeto e as soluções existentes

A principal diferença do Hermeto para as soluções existentes é o fato de se inspirar nos conceitos vivenciados nos projetos open source e tentar aplicá-los em um domínio musical. O ambiente de colaboração propriamente dito ser uma engine de wikis, a utilização de um sistema de track de tarefas para o gerenciamento do projeto, o versionamento das músicas, tudo isso se soma para criar um ambiente colaborativo nos moldes do open source.

Além disso, a própria filosofia por trás do mesmo é importada, como o fato de ser totalmente acessível a quem tiver interesse, desde o acesso aos projetos como usuário até o próprio código fonte do sistema, disponibilizado nos termos da licença GPL.

7.2 REQUISITOS

Para a presente prova de conceito definiu-se alguns requisitos que permitiriam uma experiência simples na forma de colaboração e que, como é o objetivo principal do trabalho, aproximassem o

open source e a música de uma maneira consistente.

Para isso os seguintes requisitos foram levantados:

- Permitir a criação de usuários (compositores).
- Compositores podem criar projetos de música (bandas)
- Compositores podem participar de outros projetos de música
- Cada projeto de música consiste em uma wiki.
- Um projeto de música deve permitir o tracking de tarefas a serem feitas pelos integrantes
- Um projeto de música deve permitr a edição de músicas com feedback visual
- Um projeto de música deve permitir o versionamento de suas músicas

7.3 DESIGN RATIONALE

Para atingir o objetivo do trabalho utilizou-se como peça básica o conceito de wiki. Todo o Hermeto foi construído em cima deste conceito, qual seja, edição colaborativa de conteúdo, através de uma ou mais linguagens de domínio, com o objetivo de produzir informação e conhecimento em grupo.

"A wiki is a synthesis engine. A tool in a community. It allows an individual or group (usually smallish) to discover the gaps and connections in their understandings and then through exploration of the network and comparison of the nodes, fill in meaning."(Dent, 2010)

Neste sentido o design e as ferramentas escolhidas representam um pouco esta filosofia de um engine de construção a partir da síntese. Pequenos blocos de informação combinados entre si para a realização de algo maior.

Esta é a mesma filosofia que sustenta, por exemplo, a idéia de plugins, ou arquitetura de plugins. Um plugin nada mais é do que uma pequena contribuição feita em cima de uma plataforma com o objetivo de colaborar com a comunidade de usuários, agregando novos valores para a mesma. Na prática, estes dois conceitos, wiki e plugins, irão se intersectar e até mesmo se confundir dentro do desenvolvimento do Hermeto, através do engine de wiki escolhidos, o TiddlyWiki.

7.3.1 TiddlyWiki

TiddlyWiki é um sistema de wikis implementado em javascript e html, com o diferencial de ser auto-contido em um único arquivo html. Essa característica permite, entre outras coisas, utilizá-lo como um wiki pessoal, podendo ser carregado em um pen drive, enviado por email, colocado em uma pasta compartilhada em uma rede, entre outros inúmeros possíveis usos.

Sua arquitetura, como já adiantado no tópico anterior, é totalmente baseada no conceito de extensibilidade através de plugins. Para o TiddlyWiki tudo se resume a tiddles, pequenos blocos de informação, unicamente identificáveis através de um título dentro de uma instância de wiki e que recebem significados diferentes de acordo com as tags que o demarcam. Na prática, desde um artigo de wiki tradicional, esclarecendo algum conceito ou expressão, passando pelo próprio título da wiki, suas configurações e até mesmo plugins, tudo no TiddlyWiki é um tiddle.

Esta uniformização de conceitos, bem como a utilização disto para sua extensibilidade é que o tornaram tão favoráveis como escolha para a implementação do Hermeto.

Como veremos, esta alta extensibilidade, aliada a uma comunidade atuante e uma biblioteca vasta de plugins, permitiram o reaproveitamento de muitas das funcionalidades necessárias para um sistema colaborativo nos moldes do open source, através do uso de plugins de terceiros. Foi possível, por exemplo, reaproveitar grande parte do mecanismo de gerenciamento de projetos, com abstração para tarefas, usuários e projetos, graças a uma série de plugins para o TiddlyWiki conhecida como TeamTasks.

7.3.2 Transformando uma wiki textual em uma wiki musical

Escolhida a plataforma e tendo os requisitos para a prova de conceito, o próximo passo seria transformar o TiddlyWiki, um ambiente a priori feito para um uso tradicional de wiki, em um ambiente propício à colaboração musical, tendo sempre como objetivo a aproximação de conceitos com o open source.

Para isso seria necessário fazer o que se faz quando se deseja estender alguma coisa no TiddlyWiki: criar um plugin. E como tudo no TiddlyWiki é um tiddly, o jeito de se fazer isso é criando e modificando um ou mais tiddles.

7.3.3 Criando um plugin para o TiddlyWiki

Segundo a documentação do TiddlyWiki, plugins nada mais são do que tiddlers que contém código javascript definindo uma ou mais funções que implementam a funcionalidade proposta pelo mesmo (TiddlyWiki site). Para que o TiddlyWiki reconheça tal tiddle como um plugin é necessário que o desenvolvedor marque-o com a tag systemConfig. Dessa forma ao fazer o

carregamento da página o engine de plugins do TiddlyWiki irá instalar o plugin para torná-lo disponível para outras pessoas.

Idealmente o usuário do plugin irá utilizar uma macro no formato

```
<<NomeDaMacro 'parametros'>>
```

dentro de um tiddle de sua autoria. Quando isto acontecer o TiddlyWiki irá realizar o parse da macro, verificará se a sintaxe está correta e caso positivo irá executar uma função especial implementada pelo desenvolvedor do plugin, conhecida como handler. Nesta função o desenvolvedor tem acesso, através dos parâmetros da mesma, a todo o contexto de execução da macro pelo cliente do plugin. Entre os parâmetros recebidos estão desde o título do tiddle que usou a macro, até o seu conteúdo atual, sua posição na página, entre outras informações.

Além disso, o desenvolvedor do plugin tem a sua disposição uma api que permite modificar diversos elementos da página, criar novos tiddles, modificar tiddles existentes, além de ter acesso a toda a árvore DOM do documento, já que está dentro de uma função javascript dentro de um arquivo html.

7.3.4 Linguagem Abc

Para implementar a abstração de uma edição colaborativa musical fez-se uso de uma linguagem de domínio conhecida como Abc. Originalmente desenvolvida por Chris Walshaw, como uma forma de transcrever músicas folks tradicionais para o computador, o Abc é uma linguagem para descrever músicas em formato ASCII. Esse fato histórico influenciou o seu design em diversas maneiras. Como por exemplo, o fato de que apesar de ser uma linguagem de computador o objetivo é que fosse também legível por humanos. Por outro lado, por ser uma linguagem para transcrever músicas folks tradicionais de origem principalmente do oeste europeu (Inglaterra, Escócia, Irlanda), e por tradicionalmente serem músicas simples, com apenas uma voz, o suporte a músicas com mais de uma voz ficou por muito tempo inexistente na linguagem. Hoje já é possível através de algumas extensões feitas a mesma e a própria especificação 2.0 já define esta possibilidade.

7.3.4.1 Notação

A linguagem Abc define uma gramática aonde é possível expressar tanto os próprios elementos da música, quanto elementos de metadados. O seguinte trecho é um exemplo de uma música notada em Abc (Wikipedia).

```
X:1
T:The Legacy Jig
M:6/8
L:1/8
R:jig
K:G
[|GFG BAB | gfg gab | GFG BAB | d2A AFD |
GFG BAB | gfg gab | age edB |1 dBA AFD :|2 dBA ABd |:
efe edB | dBA ABd | efe edB | gdB ABd |
efe edB | d2d def | gfe edB |1 dBA ABd :|2 dBA AFD |]
```

As primeiras seis linhas definem metadados sobre a música, como o título da mesma (T:The Legacy Jig), o tempo de compasso (M:6/8) e o tom da música (K:G).

As linhas seguintes definem as notas e suas alturas, através de letras (C = dó médio, D = ré médio, E = ré médio, ... c = dó uma oitava acima, d = ré uma oitava acima, e = mi uma oitava acima e assim por diante)

7.3.4.2 AbcJs

Por sorte, boa parte do trabalho de renderizar esta notação em uma representação gráfica fiel ao desejado pelo autor da música vem sendo realizada desde o início da linguagem através de conversores, pequenos aplicativos que pegam a entrada Abc em modo ASCII de uma canção e através de um algoritmo de conversão, transformam em uma representação gráfica em linguagens ou formatos específicos para isso. É o caso do abc2mtex, desenvolvido pelo próprio criador da linguagem em 1993 que convertia uma canção Abc em comandos TeX, utilizando-se da extensão para notação musical do mesmo, o MusicTeX. Outras implementações trataram de converter para PostScript, como é o caso do abcm2ps de Jeff's Moine e midi, como é o caso do abc2midi.

Apesar destas implementações funcionarem para uma boa gama de casos de uso, para o Hermeto não seria útil, pois todas elas são implementadas na linguagem c e a única solução para renderizar uma música através destes aplicativos seria através do envio e processamento da mesma em um servidor, o que perderia um pouco da usabilidade da colaboração.

É aí que entra uma biblioteca muito útil, denominada AbcJs. AbcJs é um conversor da linguagem

Abc para SVG, um formato de arquivo de desenho vetorial padronizada pelo W3C (World Wide Web Consortium) e aceita pela maioria dos browsers modernos através do nodo <canvas> na árvore DOM dos documentos html. E o melhor de tudo, a linguagem utilizada para realizar esta conversão é javascript, linguagem de script padrão para aplicações web client-side, implementada pela maioria dos browsers atualmente no mercado.

Utilizando esta biblioteca seria possível renderizar notação abc localmente direto no browser, sem necessiadade de nenhuma comunicação com qualquer servidor remoto. Melhoraria muito a usabilidade da colaboração, já que com ela o feedback da edição seria imeditado. Ao mesmo tempo em que o músico edita, já recebe a resposta da renderização em tempo real.

Escolhida as ferramentas, o próximo passo seria implementar um plugin que através das mesmas possibilitasse a edição de música no contexto colaborativo de TiddlyWiki.

7.3.5 Implementação de um plugin para TiddlyWiki renderizar notação Abc através da biblioteca AbcJs

A implementação do plugin se daria da seguinte maneira. Como tudo em TiddlyWiki é um tiddle, inclusive um plugin, precisaria ser criado um tiddler, que ao ser anotado com a tag systemConfig, seria carregado no load do documento e instalada no sistema de plugins do TiddlyWiki. Como dito anteriormente, o sistema de plugins do TiddlyWiki espera que o desenvolvedor defina uma função handler, que irá tratar a inclusão, por parte dos usuários do plugin, de uma macro que o desenvolvedor irá definir. Ou seja, quando um usuário deseja se utilizar do plugin ele irá escrever esta macro, dentro de um tiddler de sua autoria, passando os parâmetros necessários para o funcionamento do plugin.

Foi o que foi feito. Definiu-se uma função handler que iria chamar o código da biblioteca AbcJs passando como parâmetro a música definida pelo usuário em notação Abc. Esta função handler foi instalada dentro de um tiddler anotado com a tag systemConfig. O código deste tiddler se encontra a seguir, bem como a explicação para o mesmo:

```
//{{{

config.macros.abc = {

  handler: function (place, macroName, params, wikifier, paramString, tiddler)
  {
     var tokens = params[0];
     var el = createTiddlyElement( place, 'div', '', 'abc', '');
     abcTiddle.renderAbc(el, tokens);
```

```
  }
};

//}}}
```

A documentação do TiddlyWiki recomenda que para evitar conflito de nomes e melhor organização da estrutura de escopos do documento, o desenvolvedor de um plugin defina sua função handler dentro do objeto config.macros adicionando um objeto que irá ser o domínio do plugin no documento, neste caso 'abc'.

É possível recuperar o texto abc da música digitada pelo usuário através do primeiro parâmetro do array 'params'. Após isso, é criado um novo elemento html 'div' e inserí-lo como filho do argumento 'place', que neste caso é o tiddler em que o usuário acionou o plugin. A seguir, este novo elemento 'div' criado é enviado para uma função implementada no objeto denominado abcTiddle, aonde através da biblioteca AbcJs, o texto abc passado pelo usuário é renderizado.

Este objeto abcTiddle é uma instância global do documento definida anteriormente em uma sessão reservada para que plugins adicionem seus scripts javascripts ou de terceiros. Esta sessão é denominada pelo engine do TiddlyWiki de MarkupPostHead. por ser localizada no final da tag <head> do documento html. Um desenvolvedor de plugins pode inserir seus scripts nesta sessão através de um tiddler especial de sistema, denominado com este mesmo: MarkupPostHead.

Pois foi neste tiddler que todas as bibliotecas necesárias para a implementação do plugin foram inseridas, entre elas a biblioteca AbcJs e Raphael para a renderização do formato Abc e a biblioteca Prototype, de uso geral, mas utilizada também pela AbcJs. Abaixo se encontra listado o conteúdo deste tiddler, na implementação do plugin:

```
//biblioteca de uso geral, utilizada tanto pelo Hermeto quanto pelo AbcJs
<script src="./javascripts/prototype.js" type="text/javascript"></script>

//biblioteca de renderização, utilizada pelo AbcJs
<script src="./javascripts/raphael.js" type="text/javascript"></script>

//arquivos da biblioteca AbcJs
<script src="./javascripts/abc_tune.js" type="text/javascript"></script>
<script src="./javascripts/abc_graphelements.js" type="text/javascript"></script>
<script src="./javascripts/abc_parse_header.js" type="text/javascript"></script>
<script src="./javascripts/abc_glyphs.js" type="text/javascript"></script>
<script src="./javascripts/abc_layout.js" type="text/javascript"></script>
<script src="./javascripts/abc_tokenizer.js" type="text/javascript"></script>
```

```
<script src="./javascripts/abc_tunebook.js" type="text/javascript"></script>
<script src="./javascripts/abc_parse.js" type="text/javascript"></script>
<script src="./javascripts/abc_write.js" type="text/javascript"></script>
<script src="./javascripts/sprintf.js" type="text/javascript"></script>
<script src="./javascripts/play_embedded.js" type="text/javascript"></script>
<script src="./javascripts/wav_generator.js" type="text/javascript"></script>

//arquivo implementado pelo plugin para fazer a integração entre o handler TiddlyWiki e a biblioteca AbcJs
<script src="./javascripts/abc_tiddle.js" type="text/javascript"></script>

/*
*  Conteúdo do arquivo abc_tiddle.js que integra o handler chamado pelo TiddlyWiki com o
* código
* da biblioteca AbcJs
*/

/*
*       Objeto global utilizado pelo handler do plugin
*/
var abcTiddle = {};

/*
*       Chamada para o código do AbcJs. Recebe como parâmetro o elemento no qual
*       a música será renderizada e a o texto abc digitado pelo usuário do plugin
*/

abcTiddle.renderAbc = function (element,notation) {

  try {
        var paper = Raphael(element, 1000, 1000 );
        var tunebook = new AbcTuneBook(notation);
        var abcParser = new AbcParse();
        abcParser.parse(tunebook.tunes[0].abc);
        var tune = abcParser.getTune();
        var printer = new ABCPrinter(paper);
        printer.printABC(tune);

  } catch (e) {
        abc_plugin_errors+=e;
        console.log(e);
  }
}
```

Abaixo são listadas imagens do plugin em ação em dois momentos, ao editar e ao renderizar o resultado.

Figura 1: editando as notas através da linguagem Abc.

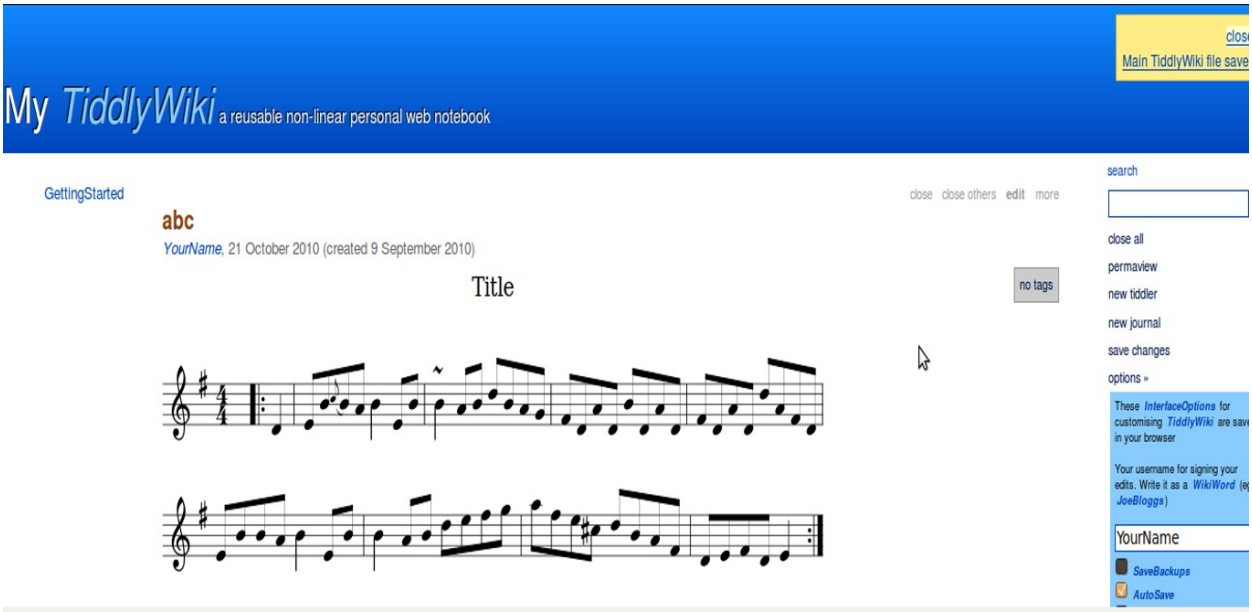

Figura 2: visualizando o resultado

7.3.6 TiddlyWiki e colaboração concorrente

Como falado anteriormente, o TiddlyWiki é um engine de wikis, auto-contido em um arquivo html, utilizado normalmente como um wiki pessoal. Este é o seu maior caso de uso, servir como um caderno de anotações pessoal, podendo ser carregado em pen drives, computadores pessoais e emails, mas sempre com o requisito de uma edição não-concorrente.

Até o slogan do mesmo reforça esta idéia: "A reusable non-linear personal web notebook". Ou seja, o suporte a edição concorrente no TiddlyWiki é inexistente. Se o html estiver em uma pasta compartilhada na rede e duas pessoas editando este mesmo arquivo, nada garante que a edição de um não vai sobrescrever o conteúdo editado pelo outro.

Isso obviamente se torna um problema quando se deseja tornar o TiddlyWiki um ambiente de colaboração realmente completo, com suporte a usuários editando concorrentemente a mesma base de dados. Para resolver este problema muitas usuários e desenvolvedores vem implementando alternativas, algumas mais simples, outras mais complexas porém mais robustas.

7.3.6.1 TiddlyWeb

A solução utilizada neste trabalho foi uma implementação de um versão servidor para o TiddlyWiki chamado TiddlyWeb. Segundo sua documentação, "TiddlyWeb é um sistema web de armazenamento de tiddlers, totalmente open source. Provê uma api http robusta, onde cada dado armazenado é uma pequena unidade de micro conteúdo endereçado por um endereço http. Entre outras coisas, funciona como uma implementação server-side para o TiddlyWiki." (documentação do TiddlyWeb)

Na nomenclatura TiddlyWeb uma coleção de tiddlers é chamada de bag, a qual se provê autorização e domínio de nomes. Bags podem ser combinadas com filtros, que através de uma receita, na nomenclatura TiddlyWiki conhecida como recipe, uma espécie de script que constrói uma instância de um TiddlyWiki juntando um ou mais tiddlers e plugins, pode criar uma vasta gama de combinações de dados, conteúdo e código.

Toda a api do TiddlyWeb é baseada no conceito de REST, ou seja, "Representational State Transfer". O termo foi cunhado por Roy Fielding em sua tese chamada Architectural Styles and the Design of Network-based Software Architectures e define uma forma de arquitetar sistemas distribuídos utilizando-se exaustivamente do protocolo HTTP para modelar recursos. Em REST

um recurso é a entidade básica disponibilizada pelo sistema e acessível pelo usuário através de diferentes representações. Um recurso é unicamente identificável através de uma uri e acessível através de urls, normalmente utilizando-se do próprio protocolo HTTP para modificar e recuperar estes recursos.

Há quatro recursos básicos disponibilizado pelo TiddlyWeb: tiddlers, bags, recipes e users. Cada um desses recursos é acessível através da seguinte api http:

* /bags
 * /bags/{bag_name}
 * /bags/{bag_name}/tiddlers
 * /bags/{bag_name}/tiddlers/{tiddler_title}
 * /bags/{bag_name}/tiddlers/{tiddler_title}/revisions
 * /bags/{bag_name}/tiddlers/{tiddler_title}/revisions/{revision}
 * /recipes
 * /recipes/{recipe_name}
 * /recipes/{recipe_name}/tiddlers
 * /recipes/{recipe_name}/tiddlers/{tiddler_title}
 * /recipes/{recipe_name}/tiddlers/{tiddler_title}/revisions
 * /recipes/{recipe_name}/tiddlers/{tiddler_title}/revisions/{revision}
 * /search

Um tiddler é a entidade básica do TiddlyWiki e pode ser desde um micro-conteúdo de informação, como um artigo de uma wiki tradicional e um post de blog, ou um código javascript que implementa uma nova funcionalidade na wiki, mais conhecido como um plugin. Bags são listas de tiddlers. Servem para agrupar tiddlers em um domínio comum, como tiddlers públicos ou privados, tiddlers internos ao sistema, entre outros. Recipes são receitas que definem como uma nova instância de um TiddlyWiki irá ser construída, que tiddlers, bags e plugins farão parte da mesma. Por fim, users nada mais são do que os usuários do sistema, identificados por login e senha.

No entanto, como falado anteriormente, o TiddlyWeb tem como objetivo funcionar como um framework que disponibiliza ao desenvolvedor maneiras de servir todos esses recursos, tradicionalmente manipulados localmente no TiddlyWiki, através da web, utilizando-se mecanismos para armazenar e recuperar estes recursos. Segundo a sua própria documentação,

"TiddlyWeb is made up of a few things:

1. A web server.

2. A data store.
3. A configuration.
4. A command line tool.

Together these things are your lego to make lots of great stuff." (documentação TiddlyWeb)

Ou seja, apesar de facilitar a criação de novas aplicações que desejam se utilizar do conceito de tiddlers e TiddlyWiki, ele por si só não serve como um "hoster" que disponibiliza a seus usuários o serviço de criação de novas instâncias de TiddlyWikis, como seria a solução ideal para o Hermeto.

7.3.7 Criando a abstração de projetos de música com TiddlySpace

É aí que então surge o TiddlySpace. Construído em cima da mesma filosofia do TiddlyWiki e do próprio TiddlyWeb, qual seja, pequenos blocos de informação, combinados com o objetivo de implementar algo maior, o TiddlySpace aparece como um plugin para o TiddlyWeb que implementa a abstração de um serviço de hoster para o mesmo. Com o TiddlySpace é possível que um usuário crie instâncias de TiddlyWikis pessoais ou para um grupo de usuários, os denominados espaços, participe colaborativamente na edição destes espaços, compartilhe tiddlers entre diferentes espaços, ou seja, fornece o que o Hermeto precisa para servir projetos de música.

Segundo sua documentação, "TiddlySpace provides a shared hosting environment for lots of users using lots of TiddlyWikis, presented in "spaces". Each space can optionally include other spaces to add content and functionality. Content can easily be followed, copied, shared, and modified to create a network of learning and sharing."

7.3.8 Gerenciando projetos de música com TeamTasks

Segundo a sua própria documentação, "TeamTasks é feito para ajudá-lo no gerenciamento de sua lista de tarefas." Nada mais é do que uma customização do TiddlyWiki que permite o gerenciamento de projetos simples, através do paradigma de "Get things done", muito utilizado em metodologias de desenvolvimento de software, inclusive em projetos open source, como descrito no capítulo IV.

No paradigma "Get things done", listas de tarefas são definidas e priorizadas e a medida que são resolvidas são marcadas como finalizadas (ou "done", em inglês).

A imagem a seguir mostra a criação de uma tarefa.

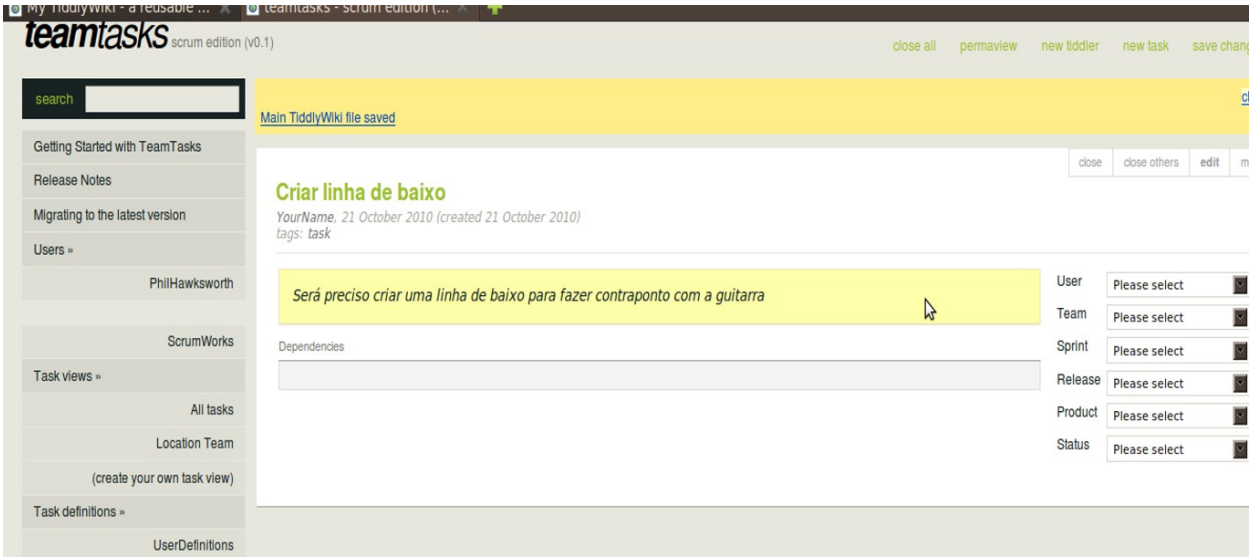

Figura 3: criando uma nova tarefa no TeamTasks

Além de tarefas, o TeamTasks permite a definição de usuários e escopos, uma espécie de delimitador de domínios. Para cada escopo são definidas tarefas configuradas com algumas propriedades. Dentre estas propriedades o criador da tarefa pode definir, por exemplo, qual é a prioridade da mesma, qual usuário ficará responsável por esta tarefa e o seu status atual.

O TeamTasks é altamente configurável, sendo possível customizar praticamente todas essas definições através de tiddlers especiais.

A imagem abaixo mostra a edição dos campos de status que uma tarefa pode ter.

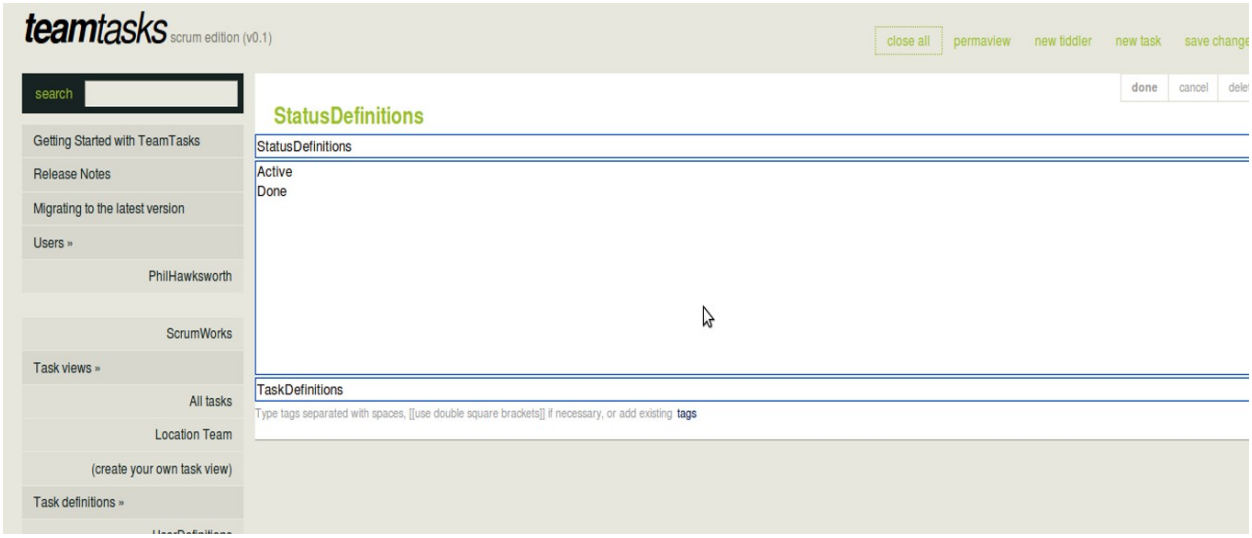

Figura 4: editando a definição de status

7.3.9 Controlando versão com RevisionPlugin

O RevisionPlugins é um plugin para o TiddlyWiki que permite gerenciar versões do conteúdo de um tiddler. É baseado no conceito de revisões, muito comum em sistemas de controle de versão de software. Nesses sistemas uma revisão é o estado de um repositório de códigos em um determinado momento de sua história. No caso do RevisionPlugin o que é versionado é o próprio conteúdo de um tiddler.

A principal feature deste plugin, além de salvar o estado de um tiddler a cada nova edição é poder visualizar estas revisões verificando exatamente o que mudou de uma para outra.

8 CONCLUSÃO

O presente trabalho teve como meta aproximar os conceitos de colaboração vivenciados pelos projetos de software aberto para um contexto de criação artística, em especial a música.

Para atingir este objetivo se fez necessário antes buscar entender o que é o open source, que fatos históricos levaram ao seu surgimento e sua sustentação como atividade produtiva viável e que motivações pessoais e sociais estão por trás do ato de contribuir com uma plataforma comum, um rossio dos tempos modernos.

Tendo isso em mente, realizou-se uma revisão histórica dos processos criativos e da função do autor em diversos momentos de sua existência, analisando as mudanças nos mecanismos e motivações da criação desde os tempos antigos até a idade média, aonde a influência da religião se fez presente, até as revoluções propostas pelo iluminismo, ao separar o místico do campo terreno, chegando aos dias de hoje, com o surgimento da internet e seu estabelecimento como agregadora de plataformas de criação colaborativa.

Dentro desse contexto, foi necessário o aprofundamento em uma dessas plataformas, o open source, do qual procurou-se extrair os mecanismos internos e as ferramentas utilizadas como forma de generalizá-las como framework para o uso em outras atividades criativas, no caso a música.

Por fim, enxergando a viabilidade da tarefa, tais generalizações puderam ser aplicadas dentro de um contexto musical, através do sistema colaborativo Hermeto.

Após tal estudo e implementação algumas conclusões puderam ser extraídas:

A primeira é que é possível sim a aproximação entre a colaboração no desenvolvimento de software com a colaboração em arte. Isso se explica pelo simples fato de que ambas são atividades criativas e muito de suas dinâmicas são muito parecidas.

Notou-se, por exemplo, que as motivações em ambas atividades giram em torno da realização pessoal e reputação no meio. Ambas são atividades que deixam rastros, que realizam e complementam a existência de quem o pratica e que por isso se diferem de um trabalho simplesmente laboral. Tanto o criador da arte quanto o desenvolvedor de software buscam incessantemente o belo no seu trabalho diário. Por isso, como atividades criativas que são, foi possível extrair mecanismos em comum entre essas duas práticas.

A segunda conclusão foi constatar que surgiram mudanças na forma de enxergar o processo criativo e a função do autor neste último meio século de história. Mudanças que tangem a visão da humanidade em relação ao artista como um criador nascido do conflito com a sociedade, que como um quase mártir consegue extrair beleza do sofrimento em uma atividade extra-humana, e

que por isso uma atividade para poucos privilegiados. Mudanças que passam a enxergar o ato criativo como algo mais natural, mais cotidiano e mais acessível para quem o deseja praticar.

E se em parte tais mudanças já vinham emergindo em atividades isoladas e movimentos de vanguarda, principalmente no meio artístico, o surgimento da internet e do estabelecimento da mesma como rede global de interações sociais bem como o barateamento dos hardwares necessários para se conectar a mesma, tiveram papel fundamental nesta transformação de paradigma ao subverter o sistema tradicional de produção artística no formato pouco para muitos, subsidiados por grandes indústrias (espécie de mecenas dos tempos modernos), para uma produção artística realmente social e democrática, no formato "faça-você-mesmo".

A terceira conclusão se refere a constatação do open source como plataforma colaborativa real e eficiente, ao empregar valores considerados humanos, como a ajuda mútua e o desejo de trabalhar em algo comum e útil, em motivações pessoais, de carreira e financeira até o desejo de se afirmar em um grupo, através de reputação.

E isso nos traz a uma conclusão importante. O open source só existe e continua existindo porque é sustentável, no sentido de que as contribuições realizadas a ele só são possíveis porque dão algo em troca a quem a realiza. Esse algo em troca pode ser desde retornos financeiros até a satisfação pessoal, mas o importante é que ele existe.

BIBLIOGRAFIA

PRETTO, Nelson De Luca. Além das redes de colaboração: internet, diversidade cultural e tecnologias do poder / Nelson De Luca Pretto, Sérgio Amadeu da Silveira : organizadores. – Salvador: EDUFBA, 2008.

PRIMO, Alex. Fases do desenvolvimento tecnológico e suas implicações nas formas de ser, conhecer, comunicar e produzir em sociedade. In Além das redes de colaboração: internet, diversidade cultural e tecnologias do poder. Salvador: EDUFBA, 2008.

LEMOS, André. Cibercultura: tecnologia e vida social na cultura contemporânea. Porto Alegre: Sulina, 2002

SIMON, Imre e VIEIRA, Miguel Said. O rossio não-rival. In Além das redes de colaboração: internet, diversidade cultural e tecnologias do poder. Salvador: EDUFBA, 2008.

FOUCAULT, Michel. O que é um autor. Lisboa: Vega, 1992.

XIMENES, Sérgio. Minidicionário da língua portuguesa. São Paulo: Ediouro, 2001.

MAFFESOLI, Michel. O mistério da conjunção: ensaios sobre comunicação, corpo e socialidade. Porto Alegre, Sulina, 2006.

BENKLER, Yochai. The wealth of networks: how social production transforms markets and freedom. Yale University Press, 2006.

BERRY, David. e MOSS, Giles. Art, Creativity, Intellectual Property and The Commons. Libre Cultura: Meditations on free culture. Editado por David M. Berry and Giles Moss. Winnipeg: Pygmalion Books , 2008.

BERRY, David. Free as in 'Free Speech' or Free as in 'Free Labour'. Libre Cultura: Meditations on free culture. Editado por David M. Berry and Giles Moss. Winnipeg: Pygmalion Books , 2008.

PAPASTERGIADIS, Nikos. The Global Need for Collaboration. Disponível em http://collabarts.org/?p=201. Última visita em 10/09/2010.

HILDEBRAND, Hermes Renato e SADDI, Liene Nunes. Uma introdução à Arte Postal. Disponível em http://www.trilhas.iar.unicamp.br/artepostal/artepostal.html. Última visita em 10/09/2010

FOGEL, Karl. Producing Open Source Software: How to Run a Successful Free Software Project. Copyright © 2005, 2006, 2007, 2008, 2009 Karl Fogel, under a CreativeCommons

STALLMAN, Richard. The Gnu Project. Disponível em http://www.gnu.org/gnu/thegnuproject.html. Última visita em 10/09/2010.

DENT, Chris. Wikiness. Disponível http://cdent.tumblr.com/post/1125897426/wikiness. Última visita em 10/09/2010.

FIELDING, Roy. Architectural Styles and the Design of Network-based Software Architectures. Acessível em http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm. Última visita em 10/09/2010

Documentação TiddlyWeb. Disponível em http://tiddlyweb-peermore.com/wiki, Última visita em 10/09/2010.

HOUAISS, Antonio et al. Dicionário Houaiss da língua portuguesa. Rio de Janeiro: Objetiva: Instituto Antônio Houaiss de Lexicografia, 2001.

OSTERLOH, Margit e ROTA, Sandra. Open Source software development – just another case of collective invention? Institute for Organization and Administrative Science: University of Zurich, 2008.

RAYMOND, E. S. The cathedral and the bazaar. O'Reilly: Sebastopol, CA, 2001

VON HIPPEL, E., G. VON KROGH. Open Source Software and the "Private-Collective" Innovation ; Model: Issues for Organization Science. Organization Science 14 (2) 209 – 223. 2003

VON HIPPEL, E. The Sources of Innovation. Oxford University Press, New York. 1988

ARENDT, H. The Human Condition. Second edition. Chicago: University of Chicago Press. 1998

A Produção Musical e os paradigmas colaborativos na Sociedade do Conhecimento. Disponível em http://conhecimento.incubadora.fapesp.br/portal/wiki/Trabalhos. Último acesso em 10/09/2010

GINSBERG, Allen. Uivo: Kadish e Outros Poemas. Porto Alegre: L&PM, 1984.

KNUTH, Donald. The art of computer programming, Volume 1. Fundamental Algorithms. Addison-Wesley, 1981.

Abc Notation, Wikipedia. Disponível em http://en.wikipedia.org/wiki/Abc_notation

Site CollabArts. Disponível em www.collabarts.org.Última visita em 10/09/2010

PRIETO, Oliver Mezquita, Open Source Software. Último acesso em 10/10/2010. Disponívelem:http://www.undisciplinedbytes.com/wpcontent/uploads/2009/10/OpenSourceSoftware.pdf

Apêndice 1: Artigo

Open Source, criação e música: uma proposta de aproximação entre o desenvolvimento de softwarre aberto e a criação musical.

Martim Azevedo do Nascimento

Departamento de Informática e Estatística
Universidade Federal de Santa Catarina (UFSC) – Florianópolis, SC – Brazil

martim00@inf.ufsc.br

Abstract. *This paper proposes an approximation of the concepts of collaborative production, experienced by open source software projects, for a musical domain. Analyzes the different factors that led to this context, to which open source is inserted, the ubiquity of networks and commons based peer production. The work explores the inner workings of open source technical infrastructure, trying to achieve widespread application in other fields such as music. Finally, presents the design and implementation of Hermeto, a musical collaboration platform inspired by patterns from the open source world, such as the use of task tracking, version control, and wikis.*

Resumo. *Este artigo propõe uma aproximação dos conceitos de produção colaborativa, vivenciados pelos projetos de software aberto, para um domínio musical. Através de pesquisa bibliográfica são analisados os diferentes fatores que levaram a este contexto, ao qual o open source se insere, de ubiquidade das redes e produção social baseada em commons. O trabalho ainda explora os mecanismos internos técnicos do open source, tentando uma generalização para a aplicação em outros domínios, como a música. Por fim, apresenta o design e a implementação do Hermeto, uma plataforma de colaboração musical inspirada nos conceitos extraídos do open source, como o uso de task tracking, controle de versões e wikis.*

1. Introdução

É notável como a humanidade modificou sua visão de arte e criação como atividade individual ao longo do tempo, oscilando entre a idéia de uma criação nascida da tradição e tendo o coletivo anônimo como principal autor, para uma visão aonde a genialidade individual do criador aparece como fator primordial.
Lemos (2002), propõe uma divisão da história aonde essas mudanças de paradigmas nos processos criativos são abordadas a partir do ponto de vista do desenvolvimento tecnológico. Segundo ele, a humanidade passou por três grandes fases aonde o desenvolvimento tecnológico teve influência nos processos de conhecimento, educação, distribuição e criação de informação. São elas a fase da indiferença, fase do conforto e fase da ubiquidade.

1.1 Fase da indiferença

A fase da indiferença, que contempla toda a pré-história e a história até o iluminismo, se apresenta como a mistura entre arte, religião, ciência e mito. O limite entre o mágico e a realidade é extremamente tênue. Nesta fase o conhecimento é um dom concedido por Deus e a tradição é um peso que se apresenta em todas as formas de expressão, onde a criação nada mais é do que a repetição pouco modificada do saber herdado ao longo da
história e creditado a um Deus.

1.2 Fase do conforto

A fase do conforto é determinada pela razão iluminista. Em contraste às trevas da idade média, o período é marcado pela separação entre ciência e religião. Neste momento a razão surge como estandarte em todos os processos de criação e conhecimento.
É nessa fase que o capitalismo se desenvolve e a produção de bens de consumo se torna o seu carro-chefe. Tais bens para serem rentáveis precisam ser raros e deprecáveis.
A arte e música também são influenciadas por este pensamento através do processo de rarificação do autor e da rivalização da obra de arte. Neste processo, o autor é transformado em um ser mítico que tem o poder de transformar a natureza e criar coisas belas. Ele é raro, porque a lógica de oferta/procura do consumo precisa que seja.
A rivalização da obra de arte se dá nesta mesma lógica do consumo. Segundo (Simon, Vieira, 2008),

"(...) é rival aquele bem ou recurso cujo uso por alguém impede (ou compete com) o uso por outra pessoa."

1.3 Fase da ubiquidade

É na terceira fase, a fase da ubiquidade, que o open source surge como fator modificante em favor de um processo criativo mais colaborativo, e possibilitado principalmente pelo surgimento de uma economia de informação conectada (networked information economy, Benkler 2006).
O indivíduo comum passa a ter a possibilidade de participar ativamente da economia global, não mais como um mero espectador ou apenas atrás de uma grande empresa como um empregado, mas sim com poderes de modificar e influenciar a produção de bens e conhecimento.

Segundo Benkler (2006, p15) foram dois os principais fatores responsáveis por essa nova economia. O primeiro grande fator foi o desenvolvimento, de um século para cá, de uma grande indústria da informação, onde os bens são intangíveis, não-rivais e facilmente digitalizados.

O outro grande fator foi o surgimento de um ambiente de comunicação construído em cima da idéia de crescimento autônomo e descentralizado, onde com um computador de baixo custo qualquer pessoa pode se conectar e participar de uma rede com alcançe global como é o caso da internet.

Esse ambiente possibilitou o aparecimento das plataformas de produção colaborativa, ou peer

production, como as comunidades open source e a própria Wikipédia. Tais plataformas demonstraram ser possível a existência de uma economia participativa, onde quem dita as regras não são realmente as grandes corporações ou governos e sim os próprios participantes, organizados em uma hierarquia baseada no mérito.

2. O Open Source

O open source surge como a primeira grande plataforma de produção social a se aproveitar das novas tecnologias e da internet.
Segundo Prieto, *"Open source is a set of principles and practices that promote access to the production and design process for various goods, products, resources and technical conclusions or advice.* (Prieto)

O termo é mais comumente utilizado no desenvolvimento de software aberto, aonde através da grande rede, pessoas distribuídas ao longo do globo colaboram em um pedaço de software, que licenciados sob os termos de contratos de uso, modificação e distribuição livre (as licenças open source), produzem soluções muitas vezes mais robustas do que sistemas proprietários.
Para se tornarem viáveis em um ambinte descentralizado e autônomo, os projetos de software aberto desenvolveram ao longo de sua existência algumas ferramentas para facilitar na colaboração remota.

Fogel (2009) define um conjunto básico de ferramentas que a maioria dos projetos open source utilizam. São elas: web site, wiki de referência, lista de emails, sistema de controle de versão, bug tracker e um canal de comunicação em tempo real, normalmente realizado através de chats ou mensagens instantâneas.
Com o objetivo de aproximar os conceitos e técnicas vivenciadas no open source, a implementação do Hermeto se baseou em três dessas ferramentas: wiki, bug tracker e sistema de controle de versões.

3. Design e implementação do Hermeto

Todo o Hermeto foi construído em cima do conceito de wikis, qual seja, edição colaborativa de conteúdo, através de uma ou mais linguagens de domínio, com o objetivo de produzir informação e conhecimento em grupo.

"A wiki is a synthesis engine. A tool in a community. It allows an individual or group (usually smallish) to discover the gaps and connections in their understandings and then through exploration of the network and comparison of the nodes, fill in meaning."(Dent, 2010)

Foi utilizado o engine de wikis TiddlyWiki. TiddlyWiki é um sistema de wikis implementado em javascript e html, com o diferencial de ser autocontido em um único arquivo html. Essa característica permite, entre outras coisas, utilizá-lo como um wiki pessoal, podendo ser carregado em um pen drive, enviado por email, colocado em uma pasta compartilhada em uma

rede, entre outros inúmeros possíveis usos.

Sua arquitetura é totalmente baseada no conceito de extensibilidade através de plugins. Para o TiddlyWiki tudo se resume a tiddles, pequenos blocos de informação, unicamente identificáveis através de um título dentro de uma instância de wiki e que recebem significados diferentes de acordo com as tags que o demarcam. Na prática, desde um artigo de wiki tradicional, esclarecendo algum conceito ou expressão, passando pelo próprio título da wiki, suas configurações e até mesmo plugins, tudo no TiddlyWiki é um tiddle.
A implementação do Hermeto consistiu no reaproveitamento de plugins existentes e na criação de um plugin para a edição colaborativa de músicas.

3.1 Criando um plugin para o TiddlyWiki

Para o TiddlyWiki, plugins nada mais são do que tiddlers que contém código javascript definindo uma ou mais funções que implementam a funcionalidade proposta pelo mesmo. Para que o TiddlyWiki reconheça tal tiddle como um plugin é necessário que o desenvolvedor marque-o com a tag systemConfig. Dessa forma ao fazer o carregamento da página o engine de plugins do TiddlyWiki irá instalar o plugin para torná-lo disponível para outras pessoas.
Idealmente o usuário do plugin irá utilizar uma macro no formato:

<<NomeDaMacro 'parametros'>>

dentro de um tiddle de sua autoria. Quando isto acontecer o TiddlyWiki irá realizar o parse da macro, verificará se a sintaxe está correta e caso positivo irá executar uma função especial implementada pelo desenvolvedor do plugin, conhecida como handler. Nesta função o desenvolvedor tem acesso, através dos parâmetros da mesma, a todo o contexto de execução da macro pelo cliente do plugin. Entre os parâmetros recebidos estão desde o título do tiddle que usou a macro, até o seu conteúdo atual, sua posição na página, entre outras informações.
Além disso, o desenvolvedor do plugin tem a sua disposição uma api que permite modificar diversos elementos da página, criar novos tiddles, modificar tiddles existentes, além de ter acesso a toda a árvore DOM do documento, já que está dentro de uma função javascript dentro de um arquivo html.

3.2 Edição colaborativa de músicas

Para implementar a abstração de edição colaborativa de música fez-se uso de uma linguagem de domínio conhecida como Abc. Originalmente desenvolvida por Chris Walshaw, como uma forma de transcrever músicas folks tradicionais para o computador, o Abc é uma linguagem para descrever músicas em formato ASCII.
A linguagem Abc define uma gramática aonde é possível expressar tanto os próprios elementos da música, quanto elementos de metadados.

```
X:1
T:The Legacy Jig
M:6/8
L:1/8
R:jig
K:G
|:GFG BAB | gfg gab | GFG BAB | d2A AFD |
GFG BAB | gfg gab | age edB |1 dBA AFD :|2 dBA ABd |:
efe edB | dBA ABd | efe edB | gdB ABd |
efe edB | d2d def | gfe edB |1 dBA ABd :|2 dBA AFD |]
```

Figura 1. Exemplo de uma música notada em Abc (Wikipedia).

Utilizando a biblioteca open source, AbcJs, foi possível fazer a renderização dessa notação direto no browser, já que a mesma faz o parse da linguagem Abc e renderiza no formato SVG, um formato de arquivo de desenho vetorial padronizada pelo W3C (World Wide Web Consortium) e aceita pela maioria dos browsers modernos através do nodo <canvas> na árvore DOM dos documentos html.

O plugin criado apenas faz a conversação entre o engine de plugins do TiddlyWiki e a api do AbcJs. O código para o plugin pode ser encontrado em http://code.google.com/p/tiddlyabcplugin/

3.3 Gerenciando projetos de música com TeamTasks

Para implementar o gerenciamento de projetos e o tracking de atividades, utilizou-se o conceito de Get Things Done, muito utilizado em projetos de software aberto. Nesta abordagem, listas de tarefas são definidas e priorizadas e a medida que são resolvidas são marcadas como finalizadas (ou "done", em inglês).
Para este fim foi utilizado uma extensão do TiddlyWiki, conhecida como TeamTasks, na qual é possível criar e gerenciar projetos, através da criação e edição de tarefas, conhecidas como tasks. Após criadas essas tarefas podem ser gerenciadas por quem está trabalhando em cima dela e por pessoas que estejam interessadas em seu atual estado.

3.4 Controlando versão com RevisionPlugin

O RevisionPlugins é um plugin para o TiddlyWiki que permite gerenciar versões do conteúdo de um tiddler. É baseado no conceito de revisões, muito comum em sistemas de controle de versão de software. Nesses sistemas uma revisão é o estado de um repositório de códigos em um determinado momento de sua história. No caso do RevisionPlugin o que é versionado é o próprio

conteúdo de um tiddler.
A principal feature deste plugin, além de salvar o estado de um tiddler a cada nova edição é poder visualizar estas revisões verificando exatamente o que mudou de uma para outra.

4. Conclusão

O presenta artigo teve como objetivo demonstrar a viabilidade de implementação de um ambiente colaborativo dentro de um domínio musical.

Baseando-se nos padrões e técnicas colaborativas surgidas dentro do open source, procurou-se extrair tais mecanismos para a aplicação em um contexto musical.

Para isso, utilizou-se do engine de wikis TiddlyWiki, reutilizando e criando extensões para o mesmo afim de implementar algumas das ferramentas utilizadas pelos projetos de software aberto, como bug tracking e controle de versão.
Por fim, conclui-se que é possível sim uma aproximação entre esses dois domínios, o desenvolvimento de software e a criação musical, tendo ambos muito o que contribuir entre si.

5. Bibliografia

LEMOS, André. Cibercultura: tecnologia e vida social na cultura contemporânea. Porto Alegre: Sulina, 2002

SIMON, Imre e VIEIRA, Miguel Said. O rossio nãorival. In Além das redes de colaboração: internet, diversidade cultural e tecnologias do poder. Salvador: EDUFBA, 2008.

BENKLER, Yochai. The wealth of networks: how social production transforms markets and freedom. Yale University Press, 2006.

PRIETO, Oliver Mezquita, Open Source Software. Último acesso em 10/10/2010. Disponívelem:http://www.undisciplinedbytes.com/wpcontent/uploads/2009/10/OpenSourceSoftware.pdf

FOGEL, Karl. Producing Open Source Software: How to Run a Successful Free Software Project. Copyright © 2005, 2006, 2007, 2008, 2009 Karl Fogel, under a CreativeCommons AttributionShareAlike (3.0) license

DENT,Chris. Wikiness. Disponível http://cdent.tumblr.com/post/1125897426/wikiness. Última visita em 10/09/2010.

Apêndice 2: fonte do plugin criado

Arquivo abc_tiddle.js

```
//    This program is free software: you can redistribute it and/or modify
//    it under the terms of the GNU General Public License as published by
//    the Free Software Foundation, either version 3 of the License, or
//    (at your option) any later version.
//
//    This program is distributed in the hope that it will be useful,
//    but WITHOUT ANY WARRANTY; without even the implied warranty of
//    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
//    GNU General Public License for more details.
//
//    You should have received a copy of the GNU General Public License
//    along with this program.  If not, see <http://www.gnu.org/licenses/>.

var abcTiddle = {};
abcTiddle.renderAbc = function (element,notation) {
        try {
                var canvas_holder = document.createElement("div");
                var paper = Raphael(element, 1000, 1000 );
                var tunebook = new AbcTuneBook(notation);
                var abcParser = new AbcParse();
                abcParser.parse(tunebook.tunes[0].abc);
                var tune = abcParser.getTune();
                var printer = new ABCPrinter(paper);
                printer.printABC(tune);
                playEmbedded = new PlayEmbedded();
                playEmbedded.play(abcParser.getTune());
        } catch (e) {
                console.log(e);
        }
}
```

Conteúdo do tiddler MarkupPostHead

```
<script src="./javascripts/prototype.js" type="text/javascript"></script>
<script src="./javascripts/raphael.js" type="text/javascript"></script>
<script src="./javascripts/abc_tune.js" type="text/javascript"></script>
<script src="./javascripts/abc_graphelements.js" type="text/javascript"></script>
<script src="./javascripts/abc_parse_header.js" type="text/javascript"></script>
```

```
<script src="./javascripts/abc_glyphs.js" type="text/javascript"></script>
<script src="./javascripts/abc_layout.js" type="text/javascript"></script>
<script src="./javascripts/abc_tokenizer.js" type="text/javascript"></script>
<script src="./javascripts/abc_tunebook.js" type="text/javascript"></script>
<script src="./javascripts/abc_parse.js" type="text/javascript"></script>
<script src="./javascripts/abc_write.js" type="text/javascript"></script>
<script src="./javascripts/sprintf.js" type="text/javascript"></script>
<script src="./javascripts/play_embedded.js" type="text/javascript"></script>
<script src="./javascripts/wav_generator.js" type="text/javascript"></script>
<script src="./javascripts/abc_tiddle.js" type="text/javascript"></script>
```

Conteúdo do tiddler abc plugin

```
//{{{
config.macros.abc = {
  handler: function (place, macroName, params, wikifier, paramString, tiddler)
  {

      var tokens = params[0];
      var el = createTiddlyElement( place, 'div', '', 'abc', '');
      abcTiddle.renderAbc(el, params[0]);

  }
};
//}}}
```