O uso de software de código aberto na construção de software proprietário

Leonardo Gomes Portella, Mario Olímpio de Menezes

Universidade Presbiteriana Mackenzie (UPM) – São Paulo – SP – Brasil

Faculdade de Computação e Informatica

leonardo.portella97@outlook.com, mario.menezes@mackenzie.br

Abstract. *This article presents an in-depth view regarding the use of open source software in the construction of private software, aiming to legitimize the use of this development model as a reference within the technology market, through bibliographic research and an exploratory case study. through the collection of information with the CTO of a company that uses the development model in question, seeking to understand how this model works in a practical situation and the onus and bonus of its use.*

Resumo. *Este artigo apresenta um estudo sobre a utilização de software de código aberto na construção de software privado, tem como objetivo relacionar um conjunto de práticas comuns no mercado no uso e interações de empresas com projetos de código aberto, identificadas e descritas na literatura com um caso real de uma empresa que faz uso de uma solução de código aberto, busca compreender neste caso específico os ônus e bônus de sua utilização.*

1. Introdução

A busca por excelência em um mercado competitivo faz com que um número crescente de empresas utilize em seus produtos de software soluções de software de código aberto, soluções de software cujo código-fonte fica disponível para que qualquer pessoa, empresa ou instituição possa estudar, utilizar ou até mesmo modificar este programa.

Esta disponibilidade de grande volume de código e soluções prontas é um grande atrativo para empresas que desejam construir soluções proprietárias e colocá-las no mercado de forma rápida, utiliza estes modelos já testados, disponíveis e em desenvolvimento por uma ampla comunidade de desenvolvedores. Análises apontam que a maior parte das empresas que utilizam software de código aberto não conhecem as diferentes abordagens para uso desta categoria de software ou não tomam decisões conscientes sobre o uso destas soluções (PWC, 2018).

Uma abordagem consciente de uso de soluções de código aberto, com o devido conhecimento das restrições de licenciamento, principalmente na construção de um produto de software proprietário é fundamental para a construção de um modelo de negócio bem-sucedido.

Este trabalho visa buscar na literatura diferentes formas com que empresas utilizam software de código aberto na construção de um software proprietário, relaciona as vantagens, desvantagens, responsabilidades e até fatores que possam impedir o seu uso para este fim, busca elicitar práticas relacionadas a forma com que empresas participam

ou usam software de código aberto, como fundações gerenciam seus projetos de código aberto, diferentes categorias de licenciamento e aspectos relacionados à dependência e manutenção. Com o intuito de localizar este estudo e não o tornar um estudo limitado e teórico, neste trabalho um caso real de uma empresa que usa software de código aberto em uma solução proprietária é descrito e suas práticas são comparadas às elicitadas nesta literatura.
Na seção seguinte são apresentados conceitos básicos que identificam o que é um software de código aberto, categorias de licenciamento e formas com que estes projetos gerenciam contribuições. Nesta seção também são apresentadas as diferentes formas de interação entre empresas e projetos de código aberto. Em seguida é apresentada a metodologia adotada neste trabalho, o detalhamento, o caso de estudo, a discussão dos resultados e limitações deste estudo e direcionamentos para estudos futuros.

2. Referencial Teórico

2.1 Projeto de Software de Código Aberto

Um projeto de software de código aberto tem como princípio a liberdade, com a garantia aos seus usuários das quatro liberdades essenciais, a possibilidade de usar, estudar, compartilhar e modificar esse software. Deste modo faz com que este software seja de grande alcance ao público, torna possível a distribuição de modo que possa ajudar o maior número de pessoas, seja pela simples utilização de um programa gratuito ou para estudos e melhorias por meio do seu código-fonte. Com a disponibilidade de seu código é possível que este projeto seja aprimorado, gera benefício assim sua comunidade de usuários e desenvolvedores por meio da geração de cópias deste projeto com versões modificadas pela sua comunidade. (BALLHAUSEN, 2019).

A liberdade é o princípio fundamental em um software livre, mas o modelo de licenciamento é muito importante, pois um dos objetivos também é preservar e manter o código livre.

De acordo com Miriam Ballhausen (2019):

> O licenciamento de software significa que os direitos de utilização do software são concedidos. Tais direitos podem ser concedidos em várias formas, incluem direitos simples/únicos de utilização como todos os outros com os mesmos privilégios ou exclusivamente para que o licenciado seja o único que pode legalmente exercer um direito particular. Uma licença pode ser territorialmente restrita (por exemplo, a União Europeia apenas) ou concedido em todo o mundo; pode ser limitado no tempo (como o que acontece com livros de capa dura antes tornam-se disponíveis livros de capa mole), ou pode ser concedida perpetuamente. Finalmente, licenças podem ser concedidas para várias categorias de utilização, incluem a reprodução no todo ou em parte; tradução, adaptação, arranjo, e outras modificações; distribuição de um programa de computador e tornar o software publicamente disponível.

O licenciamento de um software existe para regulamentar quais direitos são concedidos a utilização do programa, com sua realização em qualquer modelo utilizado para a distribuição deste software, ao se tratar de um software proprietário ou não, este licenciamento resguarda empresas, desenvolvedores e usuários.

Diversas empresas utilizam softwares de código aberto dentro de sua operação, e na criação de softwares proprietários, mas em sua grande maioria existe um desconhecimento de uma extensa cadeia de dependências que envolvem o desenvolvimento a partir de um software gratuito e a utilização do mesmo.

A cadeia de dependências, o gerenciamento de licenças, riscos e manutenção deste software a longo prazo visa através de análises e pesquisas, cerca de dois terços das empresas utilizam este modelo de negócios sem a verificação de condições de licença deste programa (PWC, 2018).

Através de um estudo sobre a esta cadeia de dependências, podemos analisar questões como o conceito de copyleft, que consiste na exigência de que qualquer programa derivado do software de código aberto em questão deve ser distribuído da mesma forma que seu software de origem.

Por este fator existem licenças mais permissivas sem copyleft, que em relação a outras, permitem que a empresa ou desenvolvedor que utilize seu software para a criação de um software proprietário possa alterar a categoria de licenciamento que deseja usar no seu produto, sem depender dos termos de licença vinculados a um software com copyleft.

Outro aspecto muito importante de um projeto de código aberto é quem controla e mantém o projeto. A continuidade, de um projeto de código aberto é um fator muito relevante para uma empresa que deseja construir uma solução proprietária sobre ele.
No quadro 1 são relacionados alguns projetos de código aberto e características importantes como a fundação responsável pelo seu controle, categoria de licença, data de estabelecimento do projeto, data de criação da fundação responsável pelo projeto e quando a fundação adotou o projeto (BUTLER et.al, 2018).

Quadro 1. Projetos de Código Aberto e suas características (Butler et al, 2019).

	Controle	**Licença**	**Data de fundação do projeto**	**Data de estabelecimento da Fundação**	**Adoção pela Fundação**
Bouncy Castle	Legion of the Bouncy Castle	MIT	2000	2013	2013
Leshan	Eclipse Foundation	EPL-1.0	2014	2004	2014
MariaDB	MariaDB Foundation	GPL-2.0	1995	1995	2012
Papyrus	Eclipse Foundation	EPL-1.0	2008	2004	2008
Solr	Apache Software Foundation	Apache-2.0	2006	1999	2006

A principal diferença entre as categorias de licença apresentados no Quadro 1 se mostra em relação a sua permissividade, seguindo os princípios das quatro

liberdades, estas licenças apresentam características como, a licença GPL é a licença de software de código aberto de maior utilização, mantendo o acesso ao seu código-fonte como pre-requisito, mas, em contrapartida, devido a esse pre-requisito não é utilizada em sofwares proprietários, visto que em seu código deve ser compartilhado, este fator implica em seu uso comercial, questão essa resolvida com a utilização da licença LGPL que se apresenta como um meio-termo entre a GPL e licenças mais permissivas. (GNU, 2007).

A licença MIT possibilita que o software que a utiliza não possua restrições de uso, podendo ser utilizado tanto em softwares de código aberto quanto em softwares proprietários, devido à inexistência de direitos do autor, desta forma sua licença pode ser alterada conforme a necessidade da utilização. (Open Source Initiative, [201-?]).

A licença Apache, tem como autor a Apache Software Foundation, podendo ser utilizada da mesma forma que a licença MIT, porem em sua implementação existe a necessidade da inclusão de aviso de direitos do autor e o consentimento com o termo de responsabilidade vinculado a sua utilização. (Apache, [201-?]).

Assim como a licença Apache, a licença EPL também se trata de uma licença de autoria de uma empresa, no caso a Eclipse Foundation, possui uma grande compatibilidade com projetos envolvendo o Eclipse, porem possui restrições em relação a sua permissividade, obrigando o desenvolvedor a licenciar patentes de suas modificações. (Eclipse, [200-?]).

Quadro 2. Governança de projetos de software de código livre (Butler et al, 2019).

	Propriedade de Ativos	**Gestão da Comunidade**	**Processos de Desenvolvimento de Software**	**Resolução de Conflitos e Mudança de Regras**	**Uso de Informações e Ferramentas**
Bouncy Castle	The Legion of the Bouncy Castle	Não documentado explicitamente	Jira	Não documentado explicitamente	Lista de discussão, Jira e GitHub
Leshan	Sierra Wireless e outros	Código de Conduta da Comunidade Eclipse	GitHub e lista de discussão do projeto	Fundação Definida	Lista de discussão, GitHub e Projeto Wiki
MariaDB	MariaDB Foundation	Código de Conduta do Ubuntu	Lista de mala direta de capitães Maria e Jira	Fundação Definida	Lista de discussão, Jira e GitHub
Papyrus	Eclipse Foundation	Código de Conduta da Comunidade Eclipse	Lista de discussão de desenvolvedores e Bugzilla	Fundação Definida	Lista de discussão, Bugzilla, Gerrit, Git, Projeto Wiki e fórum
Solr	Apache Software Foundation	Maneira Apache	Jira	Fundação Definida	Lista de discussão, Projeto Wiki e Jira

A governança de projetos de software de código livre pode melhorar a experiência e o retorno ao utilizar esta categoria de ferramenta, portanto existem praticas que devem ser seguidas ao se trabalhar com este modelo de desenvolvimento, que cada projeto tenha suas carateristas especificas, como, por exemplo, os casos apresentados no Quadro 2.

Quatro pontos podem ser elencados ao que diz respeito a observações praticas de trabalho, baseado no reporte e correção de falhas para que sua correção seja feita de maneira mais rápida, os pedidos de recursos, analisa a viabilidade deste investimento, o apoio oferecido pela comunidade através de perguntas e o compartilhamento de conhecimento e a direção de desenvolvimento, com a divisão das oportunidades entre os desenvolvedores que participam desta comunidade. (BUTLER, 2018).

Em sua cadeia de fornecimento de software, muitas empresas adotam ferramentas de código aberto devido às facilidades que são apresentadas por esse modelo, sem a devida atenção a como esta cadeia é formada, estas cadeias se baseiam em múltiplos fornecedores que suprem outros softwares de código aberto.

"Mais de 90% dos produtos de software incluem componentes de fonte aberta, a maioria dos quais são não adicionados diretamente pelos seus próprios criadores. Em vez disso, eles são uma parte inseparável das cadeias de fornecimento de software que praticamente todas as empresas dependem." (HARUTYUNYAN, 2020).

De acordo com Nikolay Harutyunyan, (2020):

> Em nossa análise de governança de código aberto, entrevistas com especialistas sugerem para a indústria melhores práticas para trabalhar com o fornecedor desde o início, para garantir a governança de origem no fornecedor do site durante o desenvolvimento de software, como oposto a uma verificação da entrega do software. Nossa análise de dados sugere que as empresas tenham uma avançada compreensão da cadeia de abastecimento FLOSS, a governança deve concentrar seus esforços em medidas preventivas, como fornecer a verificação de licença e orientação de aprovação durante a fase de desenvolvimento.

Outra questão importante trata-se principalmente do suporte oferecido a essa categoria de ferramenta, por não existir determinado contrato entre empresa e provedora dessa ferramenta, sem compensações em caso de problema técnico ou até mesmo um acordo de serviço prestado, como em diversas empresas que utilizam o conceito de SLA (Service-Level Agreement). (GUSTAVSSON, 2020).

Dentro destas problemáticas, a implementação responsável com uma abordagem completa de todos os termos que cercam a utilização de um programa de código aberto, este processo se mostra extremamente relevante para o uso estratégico desse modelo de negócio, tem como ponto de vista seus benefícios, os softwares de código aberto oferecem esses como, por exemplo, flexibilidade, agilidade, rapidez, a habilidade de testar o ambiente, uma solida segurança da informação, a maior difusão do conhecimento, os custos mais baixos e a alta capacidade de crescimento no futuro. Pode por meio desse processo contribuir para o crescimento corporativo e profissional dos indivíduos envolvidos.

2.2 Estratégias adotadas por empresas para adoção de software de código aberto

Para a adoção de um software livre em seu processo de criação e para a utilização em uma empresa, cuidados e estrategias devem sem tomados, visa que esta implementação seja vantajosa, como podemos observar no Quadro 3 (LUNDELL, 2017).

Quadro 3. Estrategias para adoção de software livre. (Björn Lundell et al, 2017).

Estratégias básicas de como uma empresa pode se envolver com projetos de código aberto.

Estratégia 1	Adote práticas de desenvolvimento aberto em contextos de empresa fechada. Isto foi referido como desenvolvimento de fonte interna.
Estratégia 2	Use ferramentas de software de código aberto no processo de desenvolvimento da própria empresa.
Estratégia 3	Use componentes de software de código aberto nos produtos de TI e de softwares que são implantados para clientes e em outros contextos de uso fora do contexto própria empresa.
Estratégia 4	Contribuir para projetos de código aberto existentes e compartilhar o desenvolvimento de seu software proprietário e lançá-los como novos projetos de código aberto: (4a) Contribuir para projetos de código aberto existentes e para os produtos de código aberto existente (que foram lançados a partir desses projetos de código aberto) (4b) Abra produtos de software que foram inicialmente desenvolvidos como softwares proprietários no contexto da própria empresa fechada e lançar esses produtos como novos projetos de código aberto.
Estratégia 5	Estabelecer relações simbióticas entre projetos de desenvolvimento fechado no contexto da empresa e projetos de código aberto estrategicamente importantes mantidos e governado fora do próprio contexto da empresa. Com um relacionamento mutuamente benéfico, estrategicamente importantes para fortalecer os negócios de uma empresa e desenvolvimento técnico. Além disso, tais relacionamentos também são benéficos para projetos de origem.

O sucesso na utilização de softwares livres está alinhado com a responsabilidade em sua adoção, com o objetivo de fortalecer esta solução na rotina de trabalho se mostra muito importante, seja por meio da utilização desta ferramenta no contexto interno da empresa quanto no alcance aos seus usuários externos.

A contribuição em comunidades mantidas para a manutenção do sistema também se torna um fator de extrema relevância, tem em vista que se constrói uma relação extremamente benéfica à empresa que utiliza esse software, pelo fato de ao seu funcionário contribuir para este projeto, seu conhecimento crescerá, desenvolve assim sua equipe de trabalho, em paralelo a isso o software também se beneficia através das melhorias implementadas.

Esta relação construída entre a empresa que utiliza e a comunidade mantedora também é interessante do ponto de vista de manutenção e correção de falhas encontradas no sistema, pode com o conhecimento obtido por essa integração adaptar e gerar melhorias a esse projeto (LUNDELL, 2017).

3. Metodologia

Neste artigo foi utilizado o método de pesquisa misto, com a abordagem de pesquisas qualitativas e quantitativas, para atingir os objetivos propostos, um levantamento bibliográfico sobre softwares de código aberto, categoria de licenciamento e governança foi realizado. Práticas e abordagens comuns de engajamento nestes projetos de empresas que constroem soluções proprietárias também foram estudadas e relacionadas.

Os critérios adotados para a seleção de artigos se baseia em artigos cuja publicação seja recente, com o foco no desenvolvimento e utilização de ferramentas de código aberto, com a utilização também referências consultadas, indicadas no índice 6.1 do artigo.

Aliado com um caso real de construção de uma solução proprietária sobre um projeto de software de código aberto é descrito e analisado segundo as características do projeto de código aberto, suas práticas de governança, estratégia de engajamento adotada pela empresa, realizado através de entrevista.

4. Resultados

4.1 Caso DOit S.A

Este estudo de caso pretende analisar um caso corporativo real, visa a investigação vantagens e desvantagens na utilização de código aberta na construção de um software proprietário, desenvolvido como um estudo de caso exploratório, foi realizada uma entrevista com o Henrique Prange, engenheiro de software e CTO na empresa DOit SA, empresa de médio porte, com sede localizada na região de Pinheiros, em São Paulo, desenvolve um sistema do tipo ERP tem como a maioria da sua base de clientes constituída por escritórios de arquitetura e interiores no Brasil, e nos Estados Unidos por armazéns farmacêuticos.

Este sistema conta com diversas ferramentas para auxiliar a rotina de trabalho, tem funcionalidades divididas em módulos, como, por exemplo:

- Financeiro: Controle total de contas a pagar e receber, emissão de boletos bancários, fluxo de caixa e demonstração de resultados (DRE).
- Projeto: Controle de despesas de projetos, detalhamento de projetos, reembolso a funcionários e-mail de cobranças a clientes e parceiros.
- Faturamento: Emissão de notas fiscais eletrônicas integrados a prefeitura do município.
- Cadastro: Banco de dados de clientes, fornecedores ou parceiros.
- Agenda: controle de calendário de equipe, uso de salas de reunião e eventos relacionados a projetos.
- E-mail: envio de e-mails em formato de mala direta.

Com a possibilidade de extração de relatórios customizáveis de praticamente todas as informações do sistema, podem ser definidas ade acordo com necessidade cada cliente.

No mercado desde 2008, a DOit desenvolve no Brasil e nos Estados Unidos sistemas de gerenciamento e controles integrados baseados em ambiente Web, com o uso uma série de ferramentas código aberto no desenvolvimento de seu “software”, como Eclipse, VSCode, Maven, PostgreSQL e Linux no deploy das aplicações. A DOit utiliza também linguagens de programação e frameworks como base de desenvolvimento como Java, Project Wonder, Quarkus, Mule e JUnit. Além destas ferramentas, a DOit também desenvolve próprias bibliotecas que são compartilhadas e utilizadas por outras empresas mundo afora, como o WOUnit e o WOInject.

Segundo Henrique, em relação às vantagens e desvantagens é possível identificar que "uma das vantagens mais evidentes do uso de ferramentas de código aberto é o fato de elas serem gratuitas.

Isso permitiu ao longo dos anos que a DOit tivesse acesso a ferramentas e frameworks sofisticados que viabilizaram o desenvolvimento do nosso produto sem custo.

Outra vantagem de usar software deste tipo, de acordo com Henrique, "a possibilidade de adaptar essas soluções para resolver problemas que não foram imaginados pelos autores originais. Seja através de "forks" ou contribuições, ou até mesmo sugestões de melhorias."

Pode ser difícil conseguir respostas ou soluções para problemas de uso em ferramentas de código aberto. A menos que exista uma comunidade de usuários engajada, a falta de um serviço de suporte que possa ser contratado para resolução de problemas pode se tornar um entrave para resolução de erros. De acordo com Henrique, "recentemente, temos visto muitas empresas por trás destes projetos mudarem as suas licenças de uso no meio do caminho".

Sendo destacado por Henrique o seguinte exemplo:

“Nós tivemos essa experiência com o Mule, onde a Mulesoft tornou diversos conectores e soluções tinham seu código aberto na versão 3.0 como closed source na versão 4.0, e cobra uma licença caríssima para migrar para a nova versão.

Ao abordar questões ligadas a comercialização deste software, para evitar problemas, não incorporamos frameworks que utilizem alguma categoria de licença copyleft – como a GPL – em nossos produtos."

Com a análise de que o produto é comercial e de código fechado, foi necessário abdicar de alguns projetos de código aberto devido a sua licença pouco permissiva.

Uma questão que também é preocupante dentro desse modelo de negócios é o abandono de software, pode prejudicar os softwares que o utilizam e sua comunidade, segundo Henrique, "o abandono de software é mais comum do que se imagina e pode afetar tanto software de código aberto quanto software proprietário. Qualquer empresa de software que está há bastante tempo no mercado vai ver, inevitavelmente, algumas de suas tecnologias se tornarem obsoletas".

No caso da DOit, é utilizado um framework proprietário chamado WebObjects cujo desenvolvimento foi abandonado há alguns anos pela Apple, está ferramenta trata-se de um sevidor de aplicação web baseado em Java que foi originalmente desenvolvido pela

NeXT Inc. empresa que em 1997 foi adquirida pela Apple, incorporada e utilizado em sistemas como MacOS Server e Xcode, removido em 2009 e descontinuado pela Apple em maio de 2016.

Felizmente, uma comunidade de desenvolvedores criou um projeto chamado Wonder para estender e corrigir problemas no WebObjects. Atualmente, a equipe de desenvolvimento da DOit é uma das maiores contribuidores desse projeto. Uma grande vantagem de projetos de código aberto nessa situação é a possibilidade de que novas pessoas possam dar continuidade ao desenvolvimento de uma ferramenta.

Ou seja, mesmo que uma empresa ou grupo de pessoas percam o interesse por um projeto, outras empresas ou grupo de pessoas pode se apropriar do desenvolvimento do projeto, portanto, trabalhar com estas ferramentas diminuem muito o risco de uma empresa ficar presa a uma tecnologia que não tem mais manutenção.

A empresa se mostra bastante preocupada com questões ligadas a governança, por portar de forma proativa em conjunto com a comunidade, pela notificação e reportação de falhas no sistema, através de fóruns de pesquisa e o GitHub se faz presente no aperfeiçoamento da ferramenta, com a consciente da importância de uma planejamento estratégico ao adicionar ferramentas de código aberto também a outras áreas de sua operação, visa contribuir para a sociedade e com seus funcionários através da difusão do conhecimento. A dependência do software de código aberto existe, mas pode ser contornada pela manutenção cíclica do seu processo de criação, como citado na entrevista, caso uma empresa ou um grupo de pessoas percam o interesse pelo projeto, novos consumidores podem se apropriar e aprimorar o que foi construído, e com a participação ativa na comunidade a implementação de bibliotecas próprias e funcionalidades específicas tornam-se mais fáceis os aprimoramentos.

5. Conclusões e Recomendações

Com a conclusão deste artigo foi possível analisar, que ao implementar o uso de softwares de código aberto no desenvolvimento de seu software proprietário a empresa deve levar em consideração questões analisadas no material pesquisado, implementar o software livre em sua operação de maneira cautelosa, com o foco desta utilização em softwares que não utilize alguma categoria de licença copyleft, para se resguardar e poder implementar seus termos de licença, com isso assegura que seu produto não sofra restrições impostas por terceiros.

No Quadro 4 são apresentadas caracterásticas do software WebOjects:

Quadro 4. Caracteristicas do WebObjects, com o uso o modelo do Quadro 1 como referência.

	Controle	**Licença**	**Data de fundação do projeto**	**Data de estabelecimento da Fundação**	**Adoção pela Fundação**
WebObjects	WOCommunity	Não documentado explicitamente	1996	2009	2016

Em paralelo com uma abordagem correta em relação à governança e a estratégia que a empresa adota através dos anos, com a participação ativa na comunidade que mantêm o software de código aberto que utiliza, valida a utilização deste modelo de desenvolvimento por meio do seu crescimento.

Em relação à governança determinada pela comunidade que gerencia o projeto WebObjects, seus pontos são apresentados no Quadro 5.

Quadro 5. Governança estabelecida pelo WebObjects, com o uso do modelo do Quadro 2 como referência .

	Propriedade de Ativos	**Gestão da Comunidade**	**Processos de Desenvolvimento de Software**	**Resolução de Conflitos e Mudança de Regras**	**Uso de Informações e Ferramentas**
WebObjects	Apple	WOCommunity Association	GitHub e lista de discussão do projeto	Fundação Definida	Projeto Wiki e GitHub

O fato da empresa não depender da comunidade para evoluir a solução, indica que sua contribuição no projeto de código aberto é efetiva, conforme a forma de trabalho descrita na estratégia 4.

De acordo com o estudo de caso realizado, a empresa DOit selecionou um projeto de código aberto com características adequadas para seu uso em uma solução proprietária, a governança adotada pela comunidade permite uma participação efetiva no desenvolvimento do projeto de código aberto e a estratégia adotada para seu uso, com capacitação de seus colaboradores para uso e colaboração no projeto se mostram adequados para a autonomia e independência da empresa, por meio deste estudo é possível identificar pontos importantes na utilização software de código aberto na construção de softwares proprietário, pontos como a flexibilidade que este modelo pode oferecer no desenvolvimento de um software, aliado com o benefício de gerar um produto rentável com a utilização de algo gratuito, ajudando pequenas e medias empresas crescerem no âmbito empresarial, é possível analisar também pontos negativos, como questões ligadas a suporte, que podem ser um entrave durante o desenvolvimento do software e a políticas de licenças, que podem gerar problemas por não serem totalmente permissíveis.

Para estudos futuros, mais casos podem ser analisados para o estabelecimento de uma metodologia para seleção de projetos de código aberto para diferentes estratégias de participação ou uso dos mesmos na construção de soluções proprietárias.

6. Referencias Bibliográficas

Apache, **APACHE LICENSES.** Disponivel em: https://www.apache.org/licenses/. Acesso em: 10 de jun. 2021.

Björn Lundell; Jonas Gamalielsson; Stefan Tengblad; Bahram Hooshyar Yousefi; Thomas Fischer; Gert Johansson, Bengt Rodung; Anders Mattsson; Johan Oppmark; Tomas Gustavsson, et al. 2017. **Addressing Lock-in, Interoperability, and Long-Term Maintenance Challenges Through Open Source: How Can Companies Strategically Use Open Source?**. In IFIP International Conference on Open Source Systems. Springer, Cham, 80--88.

Eclipse Foundation, **Eclipse Public License.** Disponivel em:https://www.eclipse.org/org/documents/epl-v10.html**.** Acesso em: 10 de jun. 2021.

GNU, **GNU GENERAL PUBLIC LICENSE.** Disponivel em: https://www.gnu.org/licenses/gpl-3.0.html. Acesso em: 10 de jun. 2021.

Miriam Ballhausen, **Free and Open Source Software Licenses Explained**. in Computer, vol. 52, no. 6, pp. 82-86, June 2019, doi: 10.1109/MC.2019.2907766.

Nikolay Harutyunyan**, Managing Your Open Source Supply Chain-Why and How?**, in Computer, vol. 53, no. 6, pp. 77-81, June 2020, doi: 10.1109/MC.2020.2983530.

Open Source Initiative, **The MIT License.** Disponivel em: https://opensource.org/licenses/mit-license.php**.** Acesso em: 10 de jun. 2021.

PWC. **Companies need a strategy for the use of Open Source Software**. Disponível em: https://www.pwc.de/en/digitale-transformation/open-source-software-management-and-compliance/companies-need-a-strategy-for-the-use-of-open-source-software.html. Acesso em: 28 de nov. 2020.

Simon Butler; Jonas Gamalielsson; Björn Lundell; Per Jonsson; Johan Sjöberg; Anders Mattsson; Niklas Rickö; Tomas Gustavsson; Jonas Feist; Stefan Landemoo and Erik Lönroth. 2018**. An investigation of work practices used by companies making contributions to established OSS projects.** In Proceedings of the 40th International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP '18). Association for Computing Machinery, New York, NY, USA, 201–210.

Tomas Gustavsson, **Managing the Open Source Dependency**, in Computer, vol. 53, no. 2, pp. 83-87, Feb. 2020, doi: 10.1109/MC.2019.2955869.

6.1 Bibliografia Recomendada

Clóvis Ricardo Montenegro de Lima; Rose Marie Santini. **PRODUÇÃO COLABORATIVA DE SOFTWARES LIVRES**: trabalho e tecnologia na sociedade da informação ". João Pessoa: Inf. & Soc.:Est., 2008.

Javier Luis Izquierdo Cánovas; Jordi Cabot. **A Survey of Software Foundations in Open Source.** Universitat Oberta de Catalunya: Barcelona (UOC), 2020.

Márcio Andrei de Oliveira Santos; Márcio José Teixeira; Dauster Souza Pereira. **Software Livre como Ferramenta de Inclusão Digital**, Faculdades Integradas de Cacoal: Rondônia, 2008;

OSSWATCH, **Open source for absolute beginners**. Disponível em: <http://osswatch.ac.uk/resources/beginners>. Acesso em: 13 set 2020.

OPENSOURCE, **What is open source?.** Disponível em:<https://opensource.com/resources/what-open-source>. Acesso em: 06 jun 2020.

Sérgio Amadeu da Silveira. **Software livre: A luta pela liberdade do conhecimento**. Editora Fundação Perseu Abramo: São Paulo, 2004.

Tim Stribos, **Understanding Open-Source and Free Software Licensing**. Disponível em:https://moqod.com/understanding-open-source-and-free-software-licensing/. Acesso em: 30 de nov. 2020.

THE ENTERPRISERS PROJECT, **8 advantages of using open source in the enterprise.** Disponível em: <https://enterprisersproject.com/article/2015/1/top-advantages-opensource-offers-over-proprietary-solutions>. Acesso em: 13 jun 2020.