UNIVERSIDADE FEDERAL FLUMINENSE
INSTITUTO DE COMPUTAÇÃO
SISTEMAS DE INFORMAÇÃO

BRENDO COSTA DOS SANTOS

OPEN INNOVATION: ESTUDO DE CASO SOBRE A INTEGRAÇÃO ESTRATÉGICA DO SOFTWARE DE CÓDIGO ABERTO E MODELOS DE NEGÓCIO DE EMPRESAS DE TI

Niterói
2024

BRENDO COSTA DOS SANTOS

OPEN INNOVATION: ESTUDO DE CASO SOBRE A INTEGRAÇÃO ESTRATÉGICA DO SOFTWARE DE CÓDIGO ABERTO E MODELOS DE NEGÓCIO DE EMPRESAS DE TI

Trabalho de conclusão de curso apresentado ao curso de Bacharelado em Sistemas de Informação, como requisito parcial para conclusão do curso. Área de concentração: Informação, Tecnologia e Inovação.

Orientador:
Prof. Dr. Flavio Luiz Seixas

Niterói
2024

Ficha catalográfica automática - SDC/BEE
Gerada com informações fornecidas pelo autor

S237o Santos, Brendo Costa dos
Open Innovation : Estudo de caso sobre a integração estratégica do software de código aberto e modelos de negócio de empresas de TI / Brendo Costa dos Santos. - 2024.
91 f.: il.

Orientador: Flavio Luiz Seixas.
Trabalho de Conclusão de Curso (graduação)-Universidade Federal Fluminense, Instituto de Computação, Niterói, 2024.

1. Código aberto (Computação). 2. Inovação aberta. 3. Negócio. 4. Produção intelectual. I. Seixas, Flavio Luiz, orientador. II. Universidade Federal Fluminense. Instituto de Computação. III. Título.

CDD - XXX

Bibliotecário responsável: Debora do Nascimento - CRB7/6368

BRENDO COSTA DOS SANTOS

OPEN INNOVATION: ESTUDO DE CASO SOBRE A INTEGRAÇÃO ESTRATÉGICA DO SOFTWARE DE CÓDIGO ABERTO E MODELOS DE NEGÓCIO DE EMPRESAS DE TI

Trabalho de conclusão de curso apresentado ao curso de Bacharelado em Sistemas de Informação, como requisito parcial para conclusão do curso. Área de concentração: Informação, Tecnologia e Inovação.

Aprovada em 22 de julho de 2024.

BANCA EXAMINADORA

Flávio Luiz Seixas

_____

Prof. Dr. Flavio Luiz Seixas - UFF

Marcelo Fornazin

_____

Prof. Dr. Marcelo Fornazin - UFF

Cosme Faria Corrêa

_____

Prof. Dr. Cosme Faria Corrêa - UFF

Niterói
2024

Dedico este trabalho à Tatiana, minha mãe, meu maior exemplo de pessoa e quem de forma inimaginavelmente amorosa sempre apoiou-me em minha jornada mostrando como a dedicação aos estudos é capaz de formar a nossa compreensão sobre a vida.

AGRADECIMENTOS

Expresso os meus mais sinceros agradecimentos ao corpo docente do Instituto de Computação da Universidade Federal Fluminense, tão essenciais para o meu processo de formação ao longo dos últimos quatro anos — parte deles marcado por um período de inúmeras incertezas quanto ao nosso futuro mas que, felizmente, foram superadas pela persistência aliada a excelência em gestão, ensino e pesquisa. Agradeço profundamente a orientação do Professor Flavio Luiz Seixas e aqueles que diretamente ou indiretamente me auxiliaram para a conclusão deste trabalho, bem como a receptividade e o comprometimento de toda a comunidade acadêmica da UFF em transformar a vida de inúmeras pessoas, incluindo a minha, através da educação.

Costumo comparar o código aberto com a ciência. A ciência tomou toda esta noção de desenvolver ideias em aberto e melhorar as ideias de outras pessoas e torná-las no que a ciência é hoje e nos incríveis avanços que tivemos. E eu comparo isso com a bruxaria e a alquimia, onde a abertura era algo que você não fazia.

Linus Benedict Torvalds

RESUMO

Para manutenção da viabilidade econômica em um contexto de alta competitividade global com custos de pesquisa e desenvolvimento crescentes aliados a menores ciclos de vida de produtos, empresas do setor de tecnologia da informação têm buscado cada vez mais abrir seus processos de inovação a fontes externas de ideias, buscando nas redes de compartilhamento e produção colaborativa de conhecimento importantes fontes de novas ideias de produtos e serviços para o mercado. O software de código aberto tem desempenhado um papel central nas empresas que buscam abrir seu processo de inovação. Entretanto, compreender como as empresas se apropriam do valor criado através de bens públicos em uma economia capitalista fortemente marcada pela concorrência demonstra ser um desafio enigmático. Este estudo tem por objetivo propor a inovação aberta como forma de condução rápida e responsiva para o processo de inovação no desenvolvimento de software. Será apresentada uma análise histórica e atual da indústria suportada por dois estudos de casos conduzidos em duas empresas brasileiras. Os resultados obtidos revelam que empresas buscam se apropriar do valor da inovação pública através de diferentes métodos como forma de se inserir e diferenciar-se no mercado. Este estudo conclui que a inovação aberta demonstra ser uma abordagem de pesquisa sólida e adequada para elucidar as razões para a integração estratégica do software de código aberto nos modelos de negócio das empresas de tecnologia da informação.

Palavras-chave: Software de Código Aberto. Inovação Aberta. Modelos de Negócio.

ABSTRACT

In order to maintain economic viability in a context of high global competitiveness with increasing research and development costs combined with shorter product life cycles, firms in the information technology sector have increasingly sought to open their innovation processes to external sources of ideas, searching in knowledge sharing and collaborative production networks for important sources of new ideas for products and services for the market. Open source software has played a central role in firms seeking to open up their innovation process. However, understanding how firms appropriate the value created through public goods in a capitalist economy strongly marked by competition proves to be an enigmatic challenge. This study aims to propose open innovation as an approach to quickly and responsively conducting the innovation process in software development. A historical and current analysis of the industry will be presented, supported by two case studies conducted with two Brazilian firms. The results obtained reveal that companies seek to appropriate the value of public innovation through different methods as a means of inserting and differentiating themselves in the market. This study concludes that open innovation proves to be a solid and appropriate research approach to elucidate the reasons for the strategic integration of open source software into the business models of information technology companies.

Keywords: Open Source Software. Open Innovation. Business Models.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

ACM	Association for Computing Machinery
AGPL	Affero General Public License
AI Lab	Artificial Intelligence Laboratory
ALGOL	Algorithmic Language
ARPANET	Advanced Research Projects Agency Network
AT&T	American Telephone and Telegraph Company
AWS	Amazon Web Services
BSD	Berkeley Software Distribution
CA	Certification Authority
CGI	Common Gateway Interface
CLOUD Act	Clarifying Lawful Overseas Use of Data Act
COBOL	Common Business Oriented Language
CONTU	Commission on New Technological Uses of Copyrighted Works
CSRG	Computer System Research Group
CVS	Concurrent Versions System
DARPA	Defense Advanced Research Projects Agency
DEC	Digital Equipment Corporation
FORTRAN	Formula Translator
FSF	Free Software Foundation
GDB	GNU Debugger
GNU	GNU's Not Unix
GPL	General Public License
HLLCA	High-level Language Computer Architecture
HP	Hewlett-Packard
HTTP	Hypertext Transfer Protocol
IBM	International Business Machines
ICP-Brasil	Infraestrutura de Chaves Públicas Brasileira
IP	Internet Protocol
ISO	International Organization for Standardization
ITS	Incompatible Timesharing System
LMI	Lisp Machines Incorporated
MIT	Massachusetts Institute of Technology
MULTICS	Multiplexed Information and Computing Service
NEC	Nippon Electric Company
OBD	On-board Diagnostics

LISTA DE ABREVIATURAS E SIGLAS

OSD	Open Source Definition
OSI	Open Source Initiative
OSI	Open Systems Interconnection
P&D	Pesquisa e desenvolvimento
PDP	Programmed Data Processor
SAGE	Semi-Automatic Ground Environment
SNA	Systems Network Architecture
SOSC	Symposium on Operating Systems Principles
SSPL	Server Side Public License
SVN	Apache Subversion
TCP	Transmission Control Protocol
TIC	Tecnologia da informação e comunicação
TV	Televisão
RCA	Radio Corporation of America
UC Berkeley	Universidade da Califórnia em Berkeley
VAX	Virtual Address Extension

SUMÁRIO

1 **INTRODUÇÃO**

O contexto econômico global moderno é caracterizado por sua alta competitividade, complexidade tecnológica e curtos ciclos de vida de produtos e serviços. Diferentemente do século XX, no qual a inovação no mercado era um processo longo e ocasional, na atualidade o rápido ritmo com o qual tecnologias são introduzidas, bem como a dinamicidade na organização do mercado e no consumo, evidenciam que as empresas já não podem mais atrelar-se aos modelos tradicionais de inovação fechada, isto é, inovar com seus próprios recursos. Organizações dependem cada vez mais de conhecimento e colaboração com indivíduos, empresas e organizações externas para manterem seu próprio processo de inovação sustentável, tomando vantagem das modernas tecnologias de informação e de rede que por si só diminuem os custos com a disseminação do conhecimento, comunicação e coordenação operacional em escala global (SAEBI; FOSS, 2015, p.4). De forma particular, empresas do setor de tecnologia da informação atualmente apostam em utilizar modelos de negócio com características abertas e colaborativas para enfrentar os desafios modernos aos seus processos de inovação (DUPARC *et al*., 2022, p. 727). O conceito de *software* de código aberto, cujas raízes remontam às décadas de 1950, ganhou força entre comunidades de desenvolvedores na década de 1980 e 1990 como uma reação contrária às práticas de mercado da época baseadas em *software* proprietários; *softwares* como o *kernel* Linux e aqueles oriundos do Projeto GNU não apenas se mostraram resilientes ao tempo e a introdução de novas tecnologias — sendo continuamente desenvolvidos há mais de três décadas — mas também foram amplamente adotados pela indústria.

Na atualidade, o conceito de código aberto pode ser encontrado no *software* com o qual o usuário doméstico interage diariamente, por exemplo, através do sistema operacional Android presente em *smartphones*, o navegador *web* Mozilla e seus derivados ou a suíte de escritório OpenOffice. Entretanto, o *software* de código aberto também está presente sob a forma de *softwares business-to-business* como os sistemas operacionais baseados no *kernel* Linux, o sistema de orquestração de contêineres Kubernetes, o servidor *web* Apache HTTP Server ou os sistemas de gerenciamento de bancos de dados MySQL e PostgreSQL — exemplos de aplicações que, embora frequentemente desconhecidas pelo usuário doméstico, fazem parte da infraestrutura de tecnologia da informação que permite a operação e continuidade de aplicações e serviços tão essenciais para a moderna economia baseada em

redes (DUPARC *et al*., 2022, p. 728). De acordo com um estudo comissionado pela Direção-Geral das Redes de Comunicação, Conteúdos e Tecnologias (DG CONNECT) da Comissão Europeia em 2018, estima-se que naquele ano as companhias situadas nos Estados da União Europeia investiram um bilhão de euros em *softwares* de código aberto, o que resultou em um impacto monetário estimado entre 65 e 95 bilhões de euros na economia europeia. Destaca-se ainda que na União Europeia, a maior parte das contribuições privadas para projetos de código aberto são oriundas de trabalhadores pertencentes a micro e pequenas empresas, em contraste com os Estados Unidos, onde a maior parte das contribuições são advindas de grandes empresas de tecnologia da informação e comunicação (BLIND *et al*., 2021, p. 14-15).

O desenvolvimento de *software* é um processo complexo naturalmente interdisciplinar, requerendo conhecimento e expertise de diversas disciplinas acadêmicas (GACEK; ARIEF, 2004, p. 34-35). O *software* de código aberto não é uma exceção à regra; é possível analisá-lo sob a ótica das disciplinas de ciência da computação, informação, sociais, gerenciais, organizacionais e direito (GACEK; ARIEF, 2004, p. 34-35). Entretanto, ainda que o *software* de código aberto seja estudado na academia há décadas, grande parte do trabalho acadêmico se dedica a explorar tópicos de natureza técnica e tecnológica relativos ao tema, consequentemente negligenciando os aspectos ditos gerenciais (DUPARC *et al*., 2022, p. 728; GRAND *et al*., 2004, p. 593). Durante a década de 2000, pesquisas sobre os aspectos gerenciais no tema, como Raymond (2001), Karels (2003), Goldman e Gabriel (2005), Bonaccorsi, Giannangeli e Rossi (2006) buscaram identificar e agrupar modelos de negócio para comercialização de *software* de código aberto a partir de reflexões teóricas sobre seu modelo de desenvolvimento e/ou análises de limitadas quantidades de casos de uso na indústria. Todavia, na atualidade, o emprego do *software* de código tornou-se uma ferramenta-padrão para o alcance de objetivos estratégicos, o que é evidenciado pelo seu crescente interesse nas pesquisas da área de sistema de informação, como em disciplinas de estratégia de plataforma e inovação aberta (DUPARC *et al*., 2022, p. 728).

Enquanto que uma ampla gama de pesquisas teóricas e empíricas sobre os aspectos técnicos do *software* de código aberto conseguem traçar as razões pelas quais empresas o incluem em seus processos de inovação interno — através de vantagens intrínsecas perceptíveis sobre *softwares* proprietários tais como qualidade, segurança, flexibilidade e custo reduzido (MORGAN; FINNEGAN, 2010, p. 83) —, as mesmas razões aparentam não responder o fluxo contrário, onde empresas contribuem com o fruto de seus esforços internos de pesquisa e desenvolvimento para projetos de *software* de código aberto. Surge então a

seguinte questão: “por que certas empresas contribuem para projetos de *software* aberto mesmo tendo a ciência de que os frutos de seus trabalhos de pesquisa e desenvolvimento muito potencialmente serão compartilhados com empresas concorrentes?”. Esse fenômeno foi descrito como “paradoxal” por Grand *et al.*, (2004), West e Gallagher (2006, p. 320) e Perr, Appleyard e Sullivan (2010, p. 433), uma vez ele desafia a lógica capitalista tradicional da inovação no mercado como um bem intelectual de propriedade exclusiva de um ou poucos agentes no mercado (MEDEIROS, 2019, p. 87-88). A emergente área de pesquisa organizacional conhecida como “inovação aberta” (“*open innovation*”) inaugurada através do influente trabalho de Chesbrough a partir da década de 2000, todavia, aparenta fornecer uma metodologia concisa baseadas em fluxos de conhecimento aliados a modelos de negócios para estudar e avaliar o fenômeno considerando o contexto social e econômico no qual as indústrias, sobretudo as de segmentos altamente tecnológicos como informação e comunicação, se inserem no século XXI.

Conforme O’Hara, Watson e Kavan (1999, p. 64) propõem, nos estudos de sistemas de informação, uma possíveis formas de se compreender a complexa interação entre tecnologias e pessoas dentro de uma organização é considerar a organização como um sistema sociotécnico no qual as organizações são vistas como a interação de quatro variáveis altamente inter-relacionadas: tarefas (ou processos), pessoas, estruturas (ou papéis), e tecnologia. Toda organização pode ser compreendida como um conjunto de partes inter-relacionadas que trabalham juntas com a finalidade de alcançar um objetivo comum. Há dois sistemas primários e interdependentes na organização: o sistema social e o sistema técnico. As pessoas realizam tarefas ou processos; essas tarefas produzem os bens e serviços da organização; a estrutura ou os papéis resultam dos sistemas de comunicação, autoridade e fluxos de trabalho que operam dentro da organização; a tecnologia refere-se a qualquer intervenção ou solução direta para a resolução de problemas, como um computador. Esperamos desenvolver uma resposta clara e concisa para o problema proposto, contribuindo assim para o estudo do código aberto para além das fronteiras tradicionais delimitadas pelo estudo da engenharia de *software*.

O movimento para o *software* de código aberto é um fenômeno contemporâneo; mais ainda é a sua expansão para a indústria de *software*. Todavia, não é possível compreender tais fenômenos em sua totalidade sem antes atentar-se aos eventos e fatores históricos que precedem seus estados atuais. O objetivo do presente trabalho é propor o uso da abordagem da inovação aberta como como forma de condução rápida e responsiva para o processo de inovação no desenvolvimento de *software* — formulando uma explicação para a seguinte

pergunta: “por que as empresas de tecnologia da informação integram o *software* de código aberto de forma estratégica aos seus negócios?”.

A formulação da pergunta de pesquisa — através do emprego explícito do advérbio interrogativo de causa “por que” —, visando entender a motivação para ocorrência de um fenômeno social bem como a contemporaneidade e o pouco controle do pesquisador sobre os eventos que controlam o fenômeno a ser estudado, segundo Yin (2001, p. 24-28), tornam o estudo de caso uma estratégia de pesquisa adequada ao trabalho.

O trabalho consiste em quatro etapas metodológicas, dispostas em seis capítulos. A etapa 1, disposta nos capítulos 1 e 2, introduz a motivação para o trabalho e seu problema de pesquisa, consistindo na elaboração de uma fundamentação teórica com base na revisão da literatura acerca da inovação aberta, sistemas de informação, gestão do conhecimento e áreas relacionadas a fim de desenvolver de forma encadeada uma linha de raciocínio composta por perspectiva histórica, atual e futura acerca do tema abordado.

A etapa 2, disposta no capítulo 3, consiste na apresentação e justificação da metodologia, bem como descrições do processo de condução dos estudos de casos realizados. A partir da fundamentação teórica, é formulada um procedimento de coleta de dados empíricos consistindo na realização de entrevistas semiestruturadas conduzidas por questionário elaborado a partir da observação de cada caso.

A etapa 3, disposta no capítulo 4, consiste na discussão, com base na fundamentação teórica desenvolvida, dos resultados obtidos através dos estudos de caso conduzidos. São apresentados os resultados e principais descobertas encontradas a partir da análise dos estudos de caso conduzidos. Os dados qualitativos foram analisados com base na fundamentação teórica conduzida na etapa 1 do presente trabalho. A análise realizada identifica e descreve as práticas de inovação aberta através do *software* de código aberto em ambas as empresas estudadas.

A etapa 4, disposta no capítulo 5 e 6, apresenta uma síntese dos resultados obtidos através do trabalho, fornecendo uma explicação sobre a questão norteadora do trabalho com base na fundamentação teórica e análise dos resultados obtidos através da condução dos estudos de caso. A explanação, conforme Yin (2001, p. 140-143) propõe, busca verificar a validade interna do trabalho através do estabelecimento de relações causais. As limitações evidenciadas na produção do trabalho são dispostas sob a forma de sugestões para trabalhos futuros.

2 **FUNDAMENTAÇÃO TEÓRICA**

O capítulo atual apresenta a fundamentação teórica do trabalho a partir de uma revisão da literatura existente. O conteúdo do capítulo é apresentado disposto em cinco subseções. A subseção 2.1 aborda historicamente as primeiras formas de desenvolvimento colaborativo de *software* presentes no período pós Segunda Guerra Mundial. A subseção 2.2 aborda historicamente o surgimento e a evolução histórica da indústria de *software* norte-americana a partir dos avanços tecnológicos das décadas de 1950 e 1960. A subseção 2.3 aborda a produção colaborativa de *software* e suas duas principais vertentes idealizando tanto o *software* quanto o seu desenvolvimento. A subseção 2.4 apresenta o conceito de inovação, bem como a maneira pela qual empresas — especialmente as de alta tecnologia — buscaram conduzir a inovação em diferentes contextos históricos. A subseção 2.5 aborda a inovação aberta através do *software* de código aberto, suas principais práticas e características e como ele atualmente se insere estrategicamente no ambiente empresarial.

2.1 **Origem da produção colaborativa de software**

De acordo com Carlotto e Ortellado (2011, p. 78), dentro dos estudos relacionados ao *software* livre e/ou de código aberto, existe um consenso acadêmico mais ou menos difundido que as práticas colaborativas para o desenvolvimento de *softwares* são anteriores ao projeto GNU e à criação da Free Software Foundation (FSF) na década de 1980. Segundo Raymond (2001, p. 69, tradução nossa[1]):

> [...] a FSF nunca foi a única iniciativa na área. Sempre existiu uma força mais silenciosa, menos confrontadora e mais amigável em relação ao mercado na cultura *hacker*. Os "pragmáticos" foram leais em menor grau a uma ideologia e em maior grau a um conjunto de tradições de engenharia fundado nos primeiros esforços do código aberto que precederam a FSF. Essas tradições incluem, mais notadamente, as culturas interligadas do Unix e da Internet pré-comercial.

[1] No original: "FSF was never the only game in town. There was always a quieter, less confrontational and more market-friendly strain in the hacker culture. The pragmatists were loyal not so much to an ideology as to a group of engineering traditions founded on early open-source efforts that predated the FSF. These traditions included, most importantly, the intertwined technical cultures of Unix and the pre-commercial Internet".

Nos anos iniciais da computação moderna, período que pode ser compreendido entre o surgimento dos primeiros computadores digitais na década de 1940 e o surgimento dos computadores pessoais na década de 1970, o *software* raramente era comercializado, isto porque grande parte do *software* neste período era fruto do trabalho de acadêmicos e engenheiros operando em universidades ou em laboratórios de corporações privadas (HIPPEL e KROGH, 2003, p. 209). Nesta época a noção do *software* como um produto de mercado não estava ainda estabelecida — prevaleciam os princípios da transparência e cooperação que regem o método científico, e o *software* era distribuído e compartilhado juntamente com seu código-fonte para ser estudado e modificado para diferentes propósitos; em partes por necessidade, uma vez em que dada a falta de arquiteturas computacionais padronizadas, *softwares* eram frequentemente incompatíveis entre diferentes computadores e sistemas operacionais. Tão logo, observou-se o surgimento das primeiras comunidades formais de usuários dedicadas ao compartilhamento de conhecimento em *software* (NOFRE, PRIESTLEY e ALBERTS, 2014 p. 55); em meados de 1955, organizações da região de Los Angeles usuárias do sistema IBM 701 da IBM (International Business Machines Corporation) tomaram ciência da incompatibilidade entre os *softwares* produzidos para o sistema e o recém anunciado IBM 704 (CERUZZI, 2003, p. 88). Paul Armer, gerente do departamento de análise numérica da RAND Corporation, Lee Amaya, gerente de computação técnica da Lockheed Aircraft, e Jack Strong e Frank Wagner, gerentes de computação e engenharia de computação da North American Aviation, se reuniram agosto daquele ano na cidade de Santa Mônica e, junto a mais 18 representantes de outras organizações, decidiram formar um grupo que logo ficou conhecido como SHARE (“compartilhar”, na língua inglesa), visando a facilitar o processo de migração de suas respectivas instalações para o novo IBM 704 (AKERA, 2001, p. 715).

Internamente, o SHARE era composto de grupos de colaboração para projetos específicos. Seus contribuidores mantinha um sistema de biblioteca de código organizado e classificado a fim de ser compartilhado e avaliado na forma de *bug reports* — registro de erros e defeitos — bem como definiam padrões de escrita de código e configuração de máquinas para facilitar a colaboração entre diferentes organizações. Através do grupo, circulavam também documentos contendo propostas e *designs* relacionados à computação (THOMAS, 2017, p. 85). O SHARE cresceu rapidamente, englobando mais empresas que contribuíram entre si para desenvolver e compartilhar rotinas operacionais, sistemas operacionais (como o *SHARE Operating System*) e um corpo de conhecimento em programação de sistemas (AKERA, 2001, p. 711). De acordo com Ceruzzi (2003, p. 88), o

SHARE não só contribui para aumentar a aceitação do IBM 704 entre organizações, como também exerceu notável influência nas decisões futuras da IBM quanto aos seus computadores e *softwares*.

2.2 **Indústria de software**

Com o decorrer dos anos, o contínuo avanço das tecnologias de *hardware*, como o surgimento da eletrônica de circuitos integrados, tornou possível sistemas computacionais cada vez mais poderosos, complexos, fisicamente menores e financeiramente baratos, contribuindo para maior adoção de sistemas computacionais por pequenas e médias organizações e posteriormente por consumidores domésticos (STEINMUELLER, 1995, p. 20). No campo do *software*, de acordo com Nofre, Priestley e Alberts (2014, p. 43-44) o surgimento dos primeiros compiladores e linguagens de programação como ALGOL e COBOL, possibilitou que o *software* fosse desvinculado de computadores específicos na medida em que atuava como uma camada de abstração entre o algoritmo e o código de máquina, o tornando portável entre diferentes sistemas computacionais (STEINMUELLER, 1995, p. 13). Nofre, Priestley e Alberts (2014), bem como Steinmueller (1995) argumentam ainda que o surgimento dessas primeiras linguagens de programação possibilitou a criação de *softwares* mais complexos destinados a executar uma gama maior de tarefas de acordo com as necessidades de seus usuários. Campbell-Kelly *et al*. (2023, p. 174) cita que o emprego da linguagem FORTRAN a partir de 1957 possibilitou programas eficientes quanto à ocupação de memória e tempo de execução. Para além disso, o emprego de FORTRAN tornou programadores mais produtivos, e programas que antes levavam dias ou semanas para serem escritos e se tornarem funcionais poderiam ser finalizados até mesmo em algumas horas (CAMPBELL-KELLY *et al*., 2023, p. 174).

Souza Júnior (2021, p. 9), defende a fundamental importância da atuação do Estado americano no fomento da nascente indústria de *software* nessa época através de investimentos em pesquisa e desenvolvimento valendo-se da complexa infraestrutura universitária do país, com destaque para o SAGE (*Semi-Automatic Ground Environment*). Considerado como o mais abrangente projeto computacional e o maior projeto computacional de larga escala da época, o SAGE operou entre 1949 e 1962, formando uma larga rede de computadores e máquinas militares dedicadas a coordenar e unificar os dados de várias estações de

equipamentos de radar a fim de formar uma única imagem do espaço aéreo de uma larga área. De acordo com Campbell-Kelly (1995, p. 82), o SAGE demandou um custo total de US$ 8 bilhões e um milhão de linhas de código, empregando 700 dos estimados 1200 programadores americanos existentes na época. Segundo Souza Júnior (2021, p. 9) o SAGE contribuiu para a capacitação de uma série de programadores para a nascente indústria de *software*, estabelecendo também os alicerces da disciplina de engenharia de *software* até então embrionária. Campbell-Kelly (1995, p. 82-83) e Souza Júnior (2021, p. 8-9) caracterizam essa indústria inicial como sendo dominada pelas grandes corporações fornecedoras de sistemas *mainframes*, pois eram as únicas com capacidade financeira e tecnológicas o suficiente para atender os principais contratos para elaboração de projetos de *software* provenientes de clientes privados e estatais de grande porte. Entretanto, essas grandes corporações não conseguiam atender o segmento composto por clientes de médio porte, uma vez que careciam de recursos para serem economicamente viáveis em negócios de menor escala. Esse vácuo no mercado seria rapidamente visto como oportunidade de negócio pelos primeiros empreendedores individuais na indústria (CAMPBELL-KELLY, 1995, p. 83).

2.2.1 Software como produto

Com a demanda crescente por *software* a partir da metade da década de 1960, a indústria passa por importantes mudanças para se manter economicamente sustentável. Embora em termos de *hardware* a capacidade computacional avançava cada vez mais, a maior demanda e o emprego maior de *softwares* nos processos operacionais empresariais tornou o processo de desenvolvimento de *software* cada vez mais complexo, ocasionando em projetos entregues com atraso, mais custosos que inicialmente orçados e permeados de erros e defeitos, fenômeno conhecido como *a crise do software* a partir do final da década (CAMPBELL-KELLY *et al*., 2023, p. 168-169). Analisando esse fenômeno, Campbell-Kelly *et al*. (2023, p. 184) reflete que as máquinas, isto é, o *hardware* físico, poderia ser fabricado em massa em fábricas; todavia, o mesmo não poderia ser aplicado ao desenvolvimento de *software*. Como consequência da crise, as corporações que detinham sistemas computacionais para seus negócios foram desestimuladas a escrever e manutenir *software* por conta própria, surgindo assim oportunidades de mercado para empresas contratantes (*contractors*) especializadas na produção de *software* "empacotado" (*packaged software*), que consistia

num só *software* concebido a atender um determinado tipo de processo ou necessidade operacional de forma genérica, sendo comercializados então a múltiplos clientes. Embora o *software* empacotado não atendesse a todas as especificidades possíveis de cada cliente — logo demandando possíveis adequações do cliente ao *software* —, o cliente possuía a vantagem de receber um produto que seria significativamente mais confiável e barato que aqueles desenvolvidos de forma autônoma e customizada (CAMPBELL-KELLY *et al*., 2023, p. 179-187). A competitividade no mercado de *software* empacotados era definida pela capacidade de desenvolvimento técnico e da comercialização de produtos em massa, necessitando portanto de elevado investimento em desenvolvimento; seu retorno financeiro era dependente da aceitação do produto pelo mercado (ANDRADE *et al*., 2009, p. 35).

De acordo com Souza Júnior (2021, p. 10), três fatores foram especialmente decisivos para a emergência dessa configuração mercadológica na indústria de *software* americana nesse período; o primeiro deles foi a popularização dos minicomputadores no mercado a partir de meados da década de 1960; os minicomputadores eram significativamente menores e mais baratos que os *mainframes*, empregando menos dispositivos de entrada e saída e menor quantidade de memória; diferentemente dos *mainframes*, não necessitavam de instalações e pessoal dedicado para sua operação, sendo operado diretamente pelo programador. Por fim, os minicomputadores dedicavam-se a controle de processos e a transmissão/retransmissão de dados em redes, enquanto que os *mainframes* dedicavam-se ao processamento e armazenamento de grandes quantidades de dados (BELL, 2014, p. 629). Destacam-se nessa época o PDP-8, lançado em 1965 pela Digital Equipment Corporation (DEC) inicialmente pelo atrativo preço de US$ 18000; tornou-se um sucesso em vendas, com mais de 50 mil unidades comercializadas. A DEC replicaria o sucesso do PDP-8 com a introdução do PDP-11 em 1970, que novamente se tornou uma linha de computadores exitosa, sendo produzido até 1990 e contando com mais de 200 mil unidades comercializadas (BELL, 2014, p. 632-633).

O segundo fator foi o sucesso comercial da família de computadores *mainframe* IBM System/360, anunciado em abril de 1964, que tornou-se a primeira plataforma padronizada na indústria de computadores (CAMPBELL-KELLY, 1995, p. 87). A família System/360 alcançou esse feito ao ser projetada tendo em mente a separação de sua arquitetura de sua implementação, permitindo que diferentes modelos de computadores compostos por variadas configurações de *hardware*, capacidade computacional e custo operacional suportassem, com certas limitações, uma só base de *software* (AMDAHL, BLAAUW e BROOKS, 1964, p. 89).

> A compatibilidade garantiria que as necessidades de expansão do usuário seriam facilmente atendidas por qualquer modelo. A compatibilidade também garantiria a máxima utilidade do sistema de suporte preparado pela fabricante, máximo compartilhamento de programas gerados pelo usuário, habilidade para usar sistemas menores como *backup* de sistemas maiores, e uma liberdade excepcional em configurar sistemas para determinadas aplicações (AMDAHL, BLAAUW e BROOKS, 1964, p. 89).

O efeito resultante desse sucesso foi elevar os custos de migração para outra plataforma, consequentemente fomentando uma ampla base instalada de computadores com o mesmo sistema operacional (SOUZA JÚNIOR, 2021, p. 10). As empresas especializadas na produção de *software* viram esse fenômeno como uma oportunidade, uma vez que um eventual produto de *software* precisaria ser escrito para uma só arquitetura com vários usuários, diminuindo assim custos com desenvolvimento e alcançando uma ampla base de consumidores (CAMPBELL-KELLY *et al*., 2023, p. 187).

O terceiro e último fator foi a decisão da IBM, anunciada em dezembro de 1968, em precificar separadamente suas soluções de negócio em *hardware* e *software*, que até então eram comercializadas conjuntamente. De acordo com Souza (2021, p. 10), a decisão foi vital para a indústria, pois "[...] fez com que o *software* deixasse de ser considerado um bem de livre acesso para ser tratado como uma mercadoria". De acordo com Wu (2020, p. 90), embora na época já houvesse algumas poucas companhias já praticando a separação em seus negócios, a decisão da IBM bem como seu grande porte e influência na indústria validou as ideias do *software* como produto e um produto lucrativo, reconceitualizando o que *software* de fato era.

2.2.2 Propriedade intelectual para software

Para Cohen (2000, p. 3) a "economia da informação" não se limita ao processo de utilizar a informação para produzir bens melhores ou executar serviços de forma mais eficaz ou eficiente, incluindo também o processo de desenvolvimento da própria informação como um produto — seja ela em forma bruta, num banco de dados ou mesmo formando as instruções de um programa de computador. A informação, antes vista como ferramenta passa a ser vista como uma *commodity* presente cada vez mais nas transações comerciais. Devido à rápida evolução e introdução de tecnologias, essas transações podem evoluir mais rapidamente que os princípios destinados a governá-las formulados pelos sistemas legais.

Todavia, não se trata em dizer que a partir de certo ponto tais transações passam a ser livres de quaisquer formas de regulamentação legal, mas sim que os sistemas legais formulados para regulamentar um contexto composto por formas de transações diferentes passam a ser forçados a regulamentar uma nova e distinta economia. Desta forma, os antigos princípios legais desenvolvidos em circunstâncias diferentes a fim de governar transações também diferentes podem não servir para governar as novas formas de transações. Os sistemas legais precisam então se ajustar frequentemente a novos contextos até que se consigam formular princípios mais a par com as novas formas de transação.

De acordo com Herscovici (2004, p. 141) o mercado não pode ser considerado como uma instância autorreguladora e socialmente eficiente, sendo necessária a presença de outros elementos não regidos pela lógica econômica para que ele possa existir e funcionar de forma satisfatória. O mercado por si só gera instabilidade nas relações entre os agentes econômicos e o sistema de preços não fornece os mecanismos adequados para anular os diferentes desequilíbrios e nem os necessários para coordenar a atuação dos agentes econômicos. "A existência de instituições, seja o Estado, as formas institucionais ou as convenções, é necessária para conter esta instabilidade e para assegurar a regulação macroeconômica" (HERSCOVICI, 2004, p. 142).

Um dos princípios que regem as transações econômicas na dita economia da informação é o instituto da propriedade intelectual. Andrade *et al*. (2009, p. 33) cita que o regime de proteção à propriedade intelectual gera importantes implicações para o processo de inovação e difusão tecnológica, abrangendo de forma multidisciplinar discussões de natureza técnica, jurídica e econômica. Medeiros (2019, p. 80) cita que se atribui à propriedade intelectual um importante papel no desenvolvimento econômico e tecnológico dos países, atribuindo a ela o papel de maximização do bem-estar social por meio da eficiência econômica provocada pela concorrência entre agentes econômicos. A proteção fornecida pela propriedade intelectual recai sobre os bens imateriais públicos, isto é, os bens que em termos econômicos são não rivais e não exclusivos, ou seja, o uso do bem por um indivíduo não reduz sua quantidade ou sua utilidade média para outrem e não impede que duas ou mais pessoas o utilizem ao mesmo tempo (MEDEIROS, 2019, p. 85). Por não serem passíveis de apropriação direta como bens materiais, os bens imateriais — sobre os quais recaem a proteção fornecida pela propriedade intelectual —, uma vez posta no mercado, podem ser reproduzidos e utilizados por qualquer indivíduo que a ele tenha acesso, extinguindo quaisquer significativos investimentos de ordem financeira, temporal e laboral que

tipicamente precedem a concepção do bem, situação que comumente caracteriza uma falha de mercado (MEDEIROS, 2019, p. 85-87).

> As falhas de mercado podem ser definidas como desvios do ótimo em relação a um sistema de preços operacionais sem custos. A existência de custos de transacção não precificados mas diferentes de zero significa que algumas transações não são criadas — transações essas que seriam realizadas caso o custo das transacções não precificadas fosse zero (ou inferior ao impacto monetário líquido a ser obtido). A falha em realizar essas negociações cria uma falha de mercado (ZERBE; MCCURDY, 1999, p. 563).

A fim de se reparar o decréscimo no bem-estar social produzido pela falha de mercado na circulação de bens imateriais institui-se então, através do Estado, o instituto da propriedade intelectual, definido no aspecto econômico por Medeiros (2019, p. 87-88) como

> [...] uma forma artificial, juridicamente criada, para corrigir essa falha de mercado, permitindo que o criador de um bem intelectual possa ter seu investimento e esforço recompensados pelo direito exclusivo de o explorar e impedir que terceiros não autorizados o façam. A exclusividade, como salienta Denis Borges Barbosa, incide no momento em que o bem imaterial adentra o mercado, tornando-se um bem-de-mercado. A propriedade intelectual é o formato adotado pelas economias de mercado e é como um direito de ordem econômica que este instituto se firma nos ordenamentos jurídicos pátrios.

Os mesmos princípios que regem a propriedade intelectual sobre bens tradicionais é aplicado também aos bens gerados na dita sociedade da informação ou sociedade informacional, caracterizada pelas novas tecnologias da informação e comunicação (TICs) (MEDEIROS, 2014, p. 477). Castells (2002, p. 67) define as TICs como sendo "o conjunto convergente de tecnologias em microeletrônica, computação (*software* e *hardware*), telecomunicação/radiodifusão, e optoeletrônica". A natureza não rival e não excludente do *software* e o crescente valor atribuído a ele na cadeia de valor das TIC a partir do início da década de 1970 representou um importante desafio para a indústria, que passou a demandar mecanismos para proteção intelectual de *software* (ANDRADE *et al*., 2009, p. 37). Embora na época já houvesse meios legais tradicionais para proteção à propriedade intelectual como o segredo industrial, a patente e o direito autoral, cada um deles fornecia um grau variado de proteção, abrangendo certos aspectos do *software* e excluindo outros (CAMPBELL-KELLY, 2005, p. 211).

Nos Estados Unidos, nesse período, as discussões sobre direitos de propriedade intelectual para *software* concentravam-se em debater se era mais adequado ou atualizar os sistemas legais para regulamentar as novas tendências tecnológicas ou ainda formular uma

regulamentação *sui generis*, com regras especificamente voltadas à proteção intelectual na área de informática (ANDRADE *et al*., 2009, p. 37). Em dezembro de 1974 o Congresso Americano criou a Comissão sobre Novos Usos Tecnológicos de Obras Protegidas por Direitos Autorais (*Commission on New Technological Uses of Copyrighted Works* — CONTU), uma comissão consultiva estabelecida para estudar formas de reformular as leis de direitos autorais até então existentes a fim de regular o uso de dispositivos tecnológicos modernos como fotocopiadoras e computadores, que possibilitam a reprodução facilitada de informações (DÍAZ, 2016, p. 753). Os legisladores esperavam que a criação do CONTU lhes permitisse deixar de lado as discussões sobre questões tecnológicas para concentrar-se nos detalhes burocráticos que impediam a aprovação do projeto de lei que se tornaria a *Copyright Act of 1976* — uma revisão legal que atua como a base primária da lei de direitos autorais dos Estados Unidos até a presente data e substituiu o antigo sistema formulado em 1909. A partir de 1974 o CONTU realizou uma extensa revisão da indústria de computação nacional, e no final de sua vigência em 1978, a comissão aconselhou o Congresso a promulgar uma legislação adicional estabelecendo o *software* como uma nova categoria de trabalho intelectual protegido pelo novo sistema legal de 1976 (DÍAZ, 2016, p. 754). A recomendação do CONTU foi aceita e em 1980 o Congresso aprovou a *Computer Software Copyright Act*, uma emenda à lei de 1976 formulada a partir dos estudos realizados pela comissão. Para fins legais, a emenda tornou o *software* elegível para estar coberto sob direitos autorais de maneira similar aos que já existiam para obras literárias (DÍAZ, 2016, p. 770-771).

2.3 **Produção colaborativa de software**

O *software* de código aberto emergiu ainda no início da década de 1980 como reação contrária ao modelo proprietário que ganhava força na época. O *software* de código aberto é precedido através de duas iniciativas distintas: a primeira foi o movimento por sistemas abertos, cujo ideal era reduzir a dependência entre consumidores e fornecedores de *software* proprietários; a segunda foi a produção de *software* através das universidades, um dos quais se destaca o *Berkeley Software Distribution* (BSD) — uma distribuição do sistema operacional Unix da AT&T desenvolvida pela Universidade da Califórnia em Berkeley no final da década de 1970. Todavia, ambos os movimentos diferem do iniciativa pelo *software* de código aberto e do movimento pelo *software* livre que viriam mais tarde, uma vez que esses últimos

focam-se majoritariamente em direitos de uso, sobretudo estabelecendo princípios que permitem o desenvolvimento colaborativo e direitos de uso perpétuos (GALLAGHER; WEST, 2006a, p. 89).

2.3.1 Adoção de sistemas abertos

O movimento por sistemas abertos surgiu dentro do próprio mercado a fim de desenvolver padrões abertos para *software*. Todavia, em contraste com outras iniciativas que viriam posteriormente, não desafiou de forma assertiva a ordem convencional da aplicação dos direitos de propriedade intelectual na indústria (HOLTGRAVE; WERLE, 2001, p. 46). O movimento consistiu em organizações fragilmente posicionadas no mercado optando por desenvolver especificações técnicas abertas a fim de atrair aliadas igualmente frágeis e majoritariamente dependentes de tecnologia proprietária. Ao abrir mão dos direitos de propriedade intelectual sobre essas especificações técnicas (nesse contexto tecnicamente denominadas "interfaces") para componentes proprietários elas permitiram que outras organizações pudessem ter acesso a mesma tecnologia, desta forma possibilitando que diversas organizações produzissem componentes interoperáveis para sistemas e redes, aumento a base de usuários desses componentes e reforçando ainda mais o processo (HOLTGRAVE; WERLE, 2001, p. 46). Destaca-se que somente as interfaces de conexão foram "abertas"; internamente os componentes permaneceram proprietários.

Cargill (1994, p. 4) divide o movimento em duas fases, denominando esse movimento inicial como "iniciativa em prol do plugue-compatível" e sendo baseado nas seguintes premissas:

- Expansão do mercado;
- Redução dos preços dos componentes de sistemas através da competição de fornecedores;
- Aumento da quantidade de fornecedores do mercado e consequentemente da competição;
- Mais responsividade do mercado às necessidades dos usuários;
- Usuários conhecerão previamente o que precisam e o que vão adquirir;
- Padrões permitirão a interoperabilidade de diferentes componentes.

Todavia, Cargill argumenta que, embora essas premissas tenham funcionado bem para sistemas unifuncionais dedicados a uma função definida tais como processamento de folhas de pagamento, contabilidade ou pedidos, elas rapidamente falharam ao serem aplicadas para sistemas mais complexos, como aqueles distribuídos, dedicados a comunicação ou bases de dados — justamente os que vinham assumindo protagonismo na computação na década de 1970 através de minicomputadores dispostos em redes. Cargill atribui a razão dessa falha às premissas, que foram formuladas tendo apenas a interconexão dos sistemas em mente, deixando de lado seus aspectos funcionais. (CARGILL, 1994, p. 4). De acordo com Holtgrave e Werle (2001, p. 46), o principal fator para a emergência da primeira fase está no fenômeno do aprisionamento tecnológico (*vendor lock-in*) que permeava a indústria computacional. Esse fator pode ser evidenciado a partir de dois subfatores: o primeiro deles era o domínio da IBM sobre o mercado de computadores que no final da década de 1970 chegava a aproximadamente 70% no segmento de *mainframes*. A introdução da família *System/360* da IBM em 1964 — previamente discutida no trabalho — trouxe para o mercado uma arquitetura que possibilitou a interoperabilidade de *software* para diferentes modelos de computadores da família. Os benefícios dessa arquitetura atraiu uma ampla base de consumidores; todavia, esses benefícios estavam limitavam-se somente à família *System/360*, o que na prática ocasionou em um aprisionamento tecnológico de consumidores aos produtos da IBM.

A segunda fase do movimento pelos sistemas abertos, denominada por Cargill (1994, p. 5) de “iniciativa em prol da intercomunicação aberta”, surgiu a partir do interesse em minicomputadores e redes de comunicação, sobretudo as baseadas em comutação de pacotes, por parte da indústria na década de 1970 — majoritariamente liderada pelas grandes corporações da indústria de eletrônica e automobilística, bem como as de serviços financeiros — que esbarrou nas dificuldades em conectar computadores e redes de fornecedores diferentes. Esse interesse foi manifestado a partir da criação de redes de comunicação desenvolvidas por iniciativas governamentais (como a ARPANET nos Estados Unidos e o CYCLADES na França) e privadas (como a *Systems Network Architecture* da IBM e a DECnet da DEC) (RUSSELL, 2014, p. 198). Organizações multinacionais lideraram a criação de redes internas globais para facilitar a troca de dados intra-organizacionalmente. As grandes corporações fornecedoras de sistemas computacionais desenvolviam *softwares* para interconectar somente as máquinas fornecidas por elas, tornando as arquiteturas das redes por si só proprietárias e incompatíveis com outras redes ou ainda máquinas de outros fornecedores. De forma similar à primeira fase, essas práticas evidenciavam a difusão do

aprisionamento tecnológico na indústria e fortaleciam ainda mais as posições de fornecedores dominantes — sobretudo a IBM (HOLTGRAVE; WERLE, 2001, p. 47).

Cargill (1994, p. 5) aponta a criação do modelo *Open Systems Interconnection* (OSI) no início da década de 1980 como um evento-chave para a segunda fase do movimento por sistemas abertos. Assim como os fornecedores de *hardware* padronizaram conexões físicas — como através do conector RS-232 e das interfaces de disco — o OSI foi criado a fim de padronizar os canais de comunicação entre computadores. O processo de padronização teve início em março de 1977 com a criação do subcomitê SC16 pela Organização Internacional de Normalização (*International Organization for Standardization* — ISO), fazendo rápido progresso durante os próximos cinco anos Russell (2014, p. 199-203).

> Graças ao seu trabalho diligente e bem divulgado, o projeto OSI tornou-se o exemplo mais notável de uma nova visão para sistemas abertos que permitia aos usuários misturar e combinar quaisquer componentes de *software* ou *hardware* que aderisse a padrões e interfaces não proprietárias. A ideologia da abertura se propôs a eliminar as fronteiras entre a grande variedade de sistemas proprietários existentes. O empoderamento do usuário decorreria naturalmente de um processo de *design* que forjava um consenso entre todas as partes interessadas em vez de ser imposto pela vontade de um único ator poderoso (RUSSELL, 2014, p. 199, tradução nossa[2]).

De acordo com Russell (2014, p. 24), "abertura foi a razão de criação do OSI, seu mais nobre fim e sua fraqueza fatal". Holtgrave e Werle (2001, p. 47) destacam a participação das grandes corporações de telecomunicações nesse processo, essas interessadas em expandir seus monopólios para a área de redes de computadores; Cargill (1994, p. 5) cita que a arquitetura do OSI foi pensada tendo em mente não só a interconexão entre sistemas, mas também suas funcionalidades, tendo reformulado as premissas da primeira fase do movimento por sistemas abertos da seguinte maneira:

- Expansão do mercado para todos os agentes nele envolvidos;
- Aumento da quantidade de fornecedores do mercado e consequentemente da competição;
- Redução dos preços dos componentes de sistemas através da competição de fornecedores;
- Mais responsividade do mercado às necessidades dos usuários;

[2] No original: "thanks to their diligent and well-publicized work, the OSI project became the most visible example of a new vision of open systems that allowed users to mix and match any software or hardware component that adhered to nonproprietary standards and interfaces. The ideology of openness proposed to eliminate boundaries among the wide variety of existing proprietary systems. User empowerment would follow naturally from a design process that forged a consensus among all interested parties rather than being imposed by the will of a single powerful actor".

- Usuários conhecerão previamente o que precisam e o que vão adquirir;
- Componentes baseados em padrões para interoperabilidade;
- Requerimento da interoperabilidade funcional entre sistemas;
- Sistemas baseados em padrões para interoperabilidade.

Embora a estrutura de padronização adotada pelo OSI tenha sido positivamente democrática, como apontam Cargill (1994, p. 5) e Russell (2014, p. 226), ela possibilitou que uma grande quantidade de organizações com diferentes visões políticas e técnicas acerca do futuro tivessem excessivo poder de participação da definição dos padrões, forçando o modelo a cobrir os mais minuciosos aspectos técnicos e consequentemente o tornando complexo e burocrático. A grande quantidade de protocolos presentes no modelo, como cita Cargill (1994, p. 5) ironicamente possibilitava que sistemas conformantes com as especificações não fossem interoperáveis entre si. Ambos os autores apontam o efeito da "padronização antecipatória" — o anseio em moldar como as futuras tecnologias deveriam ser em vez de moldar aquelas já presentes — como sendo uma das maiores razões da falha que impossibilitaram o modelo de moldar os futuros sistemas e redes abertas, como a Internet na década de 1990 — essa moldada em torno da arquitetura do TCP/IP.

2.3.2 Produção acadêmica colaborativa de software

A segunda iniciativa que precede o movimento pelo *software* de código aberto surgiu através da produção de *software* dentro dos centros de pesquisas universitários americanos. Nisso, destaca-se o desenvolvimento do sistema operacional Unix. O Unix surgiu em 1969 nos centros de pesquisa da Bell Laboratories (Bell Labs) da AT&T como um sistema operacional de tempo compartilhado, inicialmente através de um grupo de pesquisadores formado por Kenneth Thompson, Dennis M. Ritchie, Malcolm D. McIlroy e Joseph F. Ossanna. A arquitetura do sistema foi amplamente influenciada pelo MULTICS (*Multiplexed Information and Computing Service*), um projeto ambicioso de sistema operacional do qual a Bell Labs recentemente havia se retirado (RAYMOND, 2001, p. 8-9; RITCHIE, 1984, p. 1577-1593).

> O que nós queríamos preservar não era tão somente um bom ambiente para programar, mas também um sistema no qual uma comunhão pudesse ser formada ao seu entorno. Sabíamos através da experiência que a essencialidade da computação

> em comunhão, tal como suportada através de máquinas com acesso remoto e tempo compartilhado, não era só digitar programas em um terminal no lugar de uma perfuradora de cartões, mas era também encorajar a comunicação próxima (RITCHIE, 1984, p. 1577-1578).

Inicialmente escrito na linguagem de montagem do computador PDP-7 e PDP-9 da DEC, foi portado para o PDP-11 entre 1970 e 1971 e em 1973 teve seu *kernel* reescrito na linguagem de programação C recentemente concebida por Ritchie, que nela vinha trabalhando desde 1971 (RITCHIE, 1984, p. 1577-1593; RITCHIE e THOMPSON, 1974, p. 365). Inicialmente utilizado internamente nas instalações da Bell Labs para pesquisa em sistemas operacionais, linguagem de programação e redes de computadores, o sistema rapidamente atraiu interesse da comunidade acadêmica devido a sua grande portabilidade e eficiência para com e em diferentes máquinas (RITCHIE e THOMPSON, 1974, p. 365), sobretudo após ser apresentado num artigo publicado por Ritchie e Thompson na quarta edição do congresso acadêmico *Symposium on Operating Systems Principles* (SOSC), organizado pela Association for Computing Machinery (ACM), ocorrido na Universidade de Purdue em Nova York entre 15 e 17 de outubro de 1973 (SALUS, 1994, p. 58; CARLOTTO; ORTELLADO, 2011, p. 80).

A disponibilidade do sistema fora das instalações da AT&T, todavia, só foi possível devido ao acordo legal firmado entre o governo norte-americano e a empresa quase duas décadas antes, em janeiro de 1956. Naquela altura, a AT&T era a empresa dominante em serviços de telecomunicações nos Estados Unidos. Através das suas empresas operacionais, ela possuía ou controlava 98% de todas as instalações que prestavam serviços telefônicos de longa distância e 85% de todas as instalações que prestavam serviços telefônicos de curta distância no país. As empresas operadoras compravam mais de 90% dos seus equipamentos da Western Eletric Company Inc., a subsidiária de produção da AT&T. A Western Electric produzia e ofertava equipamentos de telecomunicações com base nos trabalhos de pesquisa realizados na Bell Laboratories, a empresa subsidiária de pesquisa de propriedade conjunta da AT&T e da Western Electric (WATZINGER *et al*., 2020, p. 333). Desde janeiro de 1949 o governo norte-americano, através da Divisão Antitruste do Departamento de Justiça, movia uma ação contra a AT&T e a Western Eletric pela violação do *Sherman Antitrust Act*, uma lei estabelecida em 1890 visando promover a concorrência e coibir monopólios injustificados (CARLOTTO; ORTELLADO, 2011, p. 79). O acordo firmado em 1956, sob a forma de um *consent decree*, teve como efeito a proibição da empresa de adentrar em mercados fora da indústria de telecomunicações, bem como a obrigatoriedade de licenciar todas as suas patentes tecnológicas a preços nominais e sem a cobrança de *royalties*. Como consequência do acordo,

7820 patentes da empresa ou 1,8% de todas as patentes até então legalmente protegidas nos Estados Unidos — cobrindo uma vasta gama de áreas tecnológicas — tornaram-se gratuitamente disponíveis naquela data. Grande parte dessas patentes eram frutos dos trabalhos de pesquisa desempenhados pela Bell Laboratories, responsável por grande parte da inovação tecnológica na área de telecomunicações. Todavia, mais da metade das patentes não cobriam aspectos de telecomunicações, já que o esforço científico empenhado pelos pesquisadores do laboratório durante a Segunda Guerra Mundial resultou na invenção de tecnologias referente a outras áreas da ciência, como o transistor, a célula solar, o radar, o *laser*, dentre outras tecnologias-chaves (WATZINGER *et al*., 2020, p. 329).

Impossibilitada de entrar comercialmente no mercado de *software* que crescia naquela altura, o departamento jurídico da AT&T concordou em licenciar cópias do Unix para instituições de ensino a preço nominal (SALUS, 1994, p. 56-60). Todavia, essas licenças não forneciam nenhum tipo de garantia por parte da empresa, incluindo suporte técnico. O efeito imediato foi fazer com que os usuários do sistema colaborassem entre si, compartilhando ideias, informações, programas, correções de *bugs* e *hardware* (SALUS, 1994, p. 59-65). A sexta versão do sistema, publicada em maio de 1975, tornou-se a primeira versão a ser amplamente disponibilizada fora da Bell Laboratories (TAKAHASHI; TAKAMATSU, 2013, p. 127-128). Em julho de 1975, aproximadamente 40 instituições de ensino e pesquisa contavam com instalações do Unix; em setembro de 1976 esse número havia aumentado para 138, das quais 101 estavam localizadas nos Estados Unidos; 13 no Canadá; 10 na Grã-Bretanha; 4 na Austrália; 3 em Israel; 3 nos Países Baixos; e por fim Áustria, Bélgica, Alemanha e Venezuela contando cada um com uma instituição utilizando o sistema (SALUS, 1994, p. 68).

Um dos presentes no congresso SOSC em Nova York em outubro de 1973 era Robert S. Fabry, professor da Universidade da Califórnia em Berkeley. Impressionado pela apresentação de Thompson no evento, Fabry logo uniu os departamentos de ciência da computação, estatística e matemática da universidade para realizarem uma aquisição conjunta de um computador PDP-11/45 da DEC. Ele então recebeu uma fita de Thompson contendo a quarta versão do Unix e em janeiro de 1974 o sistema foi instalado na máquina adquirida (SALUS, 1994, p. 137). O BSD surgiu no início de 1978 como uma fita magnética contendo o código-fonte para um compilador da linguagem de programação Pascal (denominado *Unix Pascal System*) e um editor de texto (o *Ex*). Aqueles interessados em obter os *softwares* arcavam apenas com o custo da fita magnética destinada a armazenar a cópia, equivalente a US$ 50 (SALUS, 1994, p. 142-143). Com a adição de mais *softwares* utilitários, em 1979 o

BSD passou de um pacote de *software* para ser distribuído contendo o Unix/32V da AT&T para a recém lançada arquitetura VAX de 32-bit da DEC (SALUS, 1994, p. 154-157). A portabilidade do Unix para arquiteturas diferentes atraiu atenção da Agência de Projetos de Pesquisa Avançada de Defesa (*Defense Advanced Research Projects Agency* — DARPA), que desejava evitar sofrer o aprisionamento tecnológico comum na indústria naquele momento (SALUS, 1994, p. 159-160). Fabry propôs à agência um contrato para desenvolver uma versão do BSD destinada a ser utilizada nos computadores que formavam a rede de computadores ARPANET da DARPA. Com um contrato de 18 meses aceito, em abril de 1980 formou-se o Computer System Research Group (CSRG) — ligado ao Departamento de Ciência da Computação da UC Berkeley — a fim de integrar o BSD aos computadores da ARPANET (MCKUSICK, 1999, p. 23). Tendo realizado esse feito com sucesso, o BSD continuou a evoluir com o passar dos anos, atendendo as necessidades da comunidade da ARPANET, incluindo uma implementação do conjunto de protocolos TCP/IP, recursos de rede, sistemas de arquivos e demais *softwares* utilitários (MCKUSICK, 1999, p. 23). A implementação nativa do TCP/IP provida pelo CSRG — em especial — provou-se um grande sucesso, uma vez que fora desenvolvida abertamente com a comunidade da ARPANET e distribuída a custo de cópia (QUARTERMAN e WILHELM, 1993, p. 33).

Todavia, por mais que o BSD fosse disponibilizado abertamente, a presença de código proprietário da AT&T requeria que quaisquer receptores do código também obtivessem uma licença da empresa. Desde novembro de 1974 a AT&T estava novamente sendo acusada de práticas monopolistas pelo governo norte-americano, o que culminou na ruptura de seu monopólio empresarial em diversas empresas menores por decisão judicial em 1982. Essas mudanças de natureza política e econômica retiraram as restrições do *consent decree* de 1956 e possibilitaram que a AT&T adentrasse no mercado de *hardware* e *software*. A partir de 1983, a AT&T internalizou o desenvolvimento do Unix, criando o Unix System Labs e alterando radicalmente a forma de distribuição e licenciamento do *software* (CARLOTTO; ORTELLADO, 2011, p. 81; SALUS, 1994, p. 189-190). Na altura em que a sétima versão do Unix foi lançada, em 1978, a licença não comercial do sistema custava em torno de US$ 7000. Uma década mais tarde, esse valor rondava cerca de US$ 100000 e em 1993 já chegava aos US$ 200000. O aumento das taxas de licenciamento do sistema por parte da AT&T deu início a um esforço gradativo do CSRG em substituir o código proprietário presente no BSD por código próprio e aberto (SALUS, 1994, p. 210 e 222-223).

2.3.3 Software livre

Os avanços corporativos para abarcar o *software* como um produto de mercado protegido pela propriedade intelectual afetou significativamente a cultura colaborativa para produção de *software* no início da década de 1980. Tomando como exemplo o desenvolvimento do BSD, Carlotto e Ortellado (2011, p. 79) citam que "a ausência de uma política de licenciamento de direitos autorais levou a uma incorporação pelo mercado que introduziu práticas competitivas e fragmentou o código". Raymond (2001, p. 70) argumenta que o dito pragmatismo das comunidades ao entorno do desenvolvimento do BSD falhou em criar um modelo de desenvolvimento auto sustentável, o tornando fragmentado.

Uma reação ao conturbado cenário da cultura colaborativa para produção de *software* nesse período foi a criação da iniciativa pelo *software* livre por Richard M. Stallman através do Projeto GNU em setembro de 1983. Stallman havia sido contratado como programador pelo Laboratório de Inteligência Artificial (*Artificial Intelligence Laboratory* — AI Lab) do Instituto de Tecnologia de Massachusetts (*Massachusetts Institute of Technology* — MIT) em 1971, tendo previamente trabalhado também nos laboratórios da IBM em Nova York (CARLOTTO; ORTELLADO, 2011, p. 82). Durante essa época Stallman teve contato com a chamada "cultura *hacker*", uma subcultura forjada pelos primeiros programadores universitários na década de 1960 baseada numa "ética *hacker*" que defendia sobretudo o acesso a computadores de forma democrática, o livre acesso e circulação da informação; a promoção da descentralização organizacional e avaliações com base em habilidades em vez de critérios baseados na formação acadêmica, idade, raça ou posição social do indivíduo (LEVY, 2010, p. 27-31). Essa subcultura poderia ser encontrada em instituições acadêmicas como o MIT, Berkeley e Stanford (CARLOTTO; ORTELLADO, 2011, p. 83).

Desde o final da década de 1970, grande parte dos *hackers* do AI Lab dedicavam-se a desenvolver o *Lisp Machine*, um computador de arquitetura em linguagem de alto nível (*high-level language computer architecture* — HLLCA), bem como seu sistema operacional, como parte de um projeto de pesquisa financiado pelo DARPA. Sendo um HLLCA para a linguagem de programação Lisp, o *Lisp Machine* tinha sua arquitetura e *hardware* especificamente projetados e otimizados para processar instruções em Lisp. A capacidade da linguagem em ser adequada para a escrita de programas complexos operando com dados de estrutura irregular inicialmente a tornou uma das mais utilizadas para a pesquisa na área de inteligência artificial (KAISLER, 1986, p. 468; WILLIAMS, 2010, p. 91-92); os *hackers* do

AI Lab procuravam então juntar os recursos do Lisp com um *hardware* específico para alto desempenho. Todavia, no início da década de 1980, dois grupos rivais de *hackers* do laboratório decidiram formar suas próprias *startups* para comercializar os frutos do projeto de pesquisa no *Lisp Machine*, a Lisp Machines Incorporated (LMI) e a Symbolics Incorporated, consequentemente esvaziando grande parte do quadro de funcionários do laboratório. Com poucos *hackers* restantes, as manutenções em programas e máquinas do laboratório passaram a levar mais tempo para serem realizadas — ou em alguns casos deixaram de ser feitas (WILLIAMS, 2010, p. 92). Quando o laboratório conseguiu recursos para substituir seu já defasado computador PDP-10 pelo DECSYSTEM-20 em 1982, a falta de pessoal impossibilitou que o *Incompatible Timesharing System* (ITS) — sistema operacional desenvolvido de forma colaborativa no laboratório — fosse portado para o novo computador, obrigando a instalação a recorrer ao TOPS-20 (informalmente conhecido como TWENEX) de código fechado e propriedade da DEC (WILLIAMS, 2010, p. 92).

Stallman observou que aquela cultura já bem estabelecida começara a desmoronar; com a introdução das políticas de propriedade intelectual para *software*, as empresas da indústria cessaram a distribuição do código-fonte — legível por humanos —, com o usuário final tendo acesso apenas ao código-objeto, formado por instruções de máquina e cuja legibilidade é difícil.

A LMI e a Symbolics, que obtiveram uma licença para comercializar o código do sistema operacional do *Lisp Machine* do AI Lab, eram legalmente obrigadas a licenciar suas modificações no código-fonte para o sistema desenvolvido no laboratório, embora os termos não obrigassem as empresas a permitir a redistribuição das mudanças para outras partes interessadas, o que foi alcançado através de um acordo informal firmado em 1981. Entretanto, a Symbolics, que inicialmente concordara em disponibilizar o código-fonte de suas modificações com o AI Lab, decidiu parar de fazê-lo em março de 1982 a fim de impedir que a LMI — empresa concorrente — obtivesse proveito delas (WILLIAMS, 2010, p. 95). A Symbolics passou então a providenciar para o laboratório somente uma cópia do código-fonte modificado sob divulgação restrita e para fins de teste. Stallman, como principal desenvolvedor do sistema do laboratório, passou a reimplementar de forma própria e exitosa as mudanças e avanços no código da Symbolics a partir de sua documentação de interfaces com o objetivo de manter ambos os sistemas a par enquanto evitava uma possível acusação de violação de direitos autorais. Enquanto esse feito consolidou sua reputação entre os membros da comunidade *hacker*, Stallman passou a ser ostracizado por antigos companheiros

empregados na Symbolics, bem como tornou-se cada vez mais descontente com a situação hostil em que se encontrava um ambiente outrora acolhedor (WILLIAMS, 2010, p. 95-99).

Em 27 de setembro de 1983, Stallman anuncia no grupo de discussão net.unix-wizards da Usenet sua intenção de escrever seu próprio sistema Unix-compatível — chamado GNU (um acrônimo recursivo na língua inglesa para "*GNU is Not Unix*" ou "GNU não é Unix") — e convocando quaisquer ajudas para contribuir no desenvolvimento do sistema. Embora o sistema não seria idêntico ao Unix, seria capaz de executar seus *softwares* usuais, bem como ser dotado de recursos não presentes nas demais implementações do Unix, como uma interface gráfica de usuário, um sistema de arquivos a prova de falhas e protocolos de redes projetados conforme o sistema de rede interno do MIT (WILLIAMS, 2010, p. 89-90).

Em janeiro de 1984, Stallman decide resignar seu posto no AI Lab para dedicar-se ao Projeto GNU, embora ainda lhe fosse permitido utilizar as instalações do laboratório. A saída de Stallman do laboratório foi necessária para manter a sobrevivência do projeto, uma vez que a nova legislação conhecida como *Bayh–Dole Act*, aprovada em dezembro de 1980, passou a garantir o direito de instituições de pesquisa em patentear inovações provenientes de pesquisas financiadas pelo governo federal; portanto, o trabalho desenvolvido por Stallman nas instalações do laboratório poderia ser apropriado pelo MIT para fins comerciais, algo do qual ele veementemente discordava (TAKAHASHI; TAKAMATSU, 2013, p. 131; WILLIAMS, 2010, p. 102-103). Pelo resto do ano, Stallman e os colaboradores do Projeto GNU empenharam-se em escrever o editor de texto GNU Emacs, o depurador de código GNU Debugger (GDB), o gerador de *parser* GNU Bison, um *linker*, um *shell*, bem como outros 35 *softwares* utilitários para compor o ecossistema do sistema operacional GNU (STALLMAN, 1985, p. 30). Em março de 1985, Stallman publica um manifesto intitulado "The GNU Manifesto" no 101º volume da revista *Dr. Dobb's Journal*, argumentando suas razões ideológicas por trás do projeto, contra-argumentando eventuais críticas ao seu modelo e objetivos e convocando a contribuição de colaboradores.

Não o bastante produzir os *softwares*, Stallman acreditava ainda que eles deveriam ser fundamentalmente livres. O *software* livre, então, deveria ser aquele *software* controlado por seus usuários em vez do contrário; esse controle seria proveniente de quatro liberdades fundamentais que caracterizariam um *software* como um *software* livre (WILLIAMS, 2010, p. 181). São essas liberdades (WILLIAMS, 2010, p. 3):

- A liberdade do usuário em poder executar o *software* como ele desejar, para quaisquer propósitos (liberdade 0);

- A liberdade do usuário em estudar como o *software* funciona, e adaptá-lo às suas necessidades (liberdade 1). Para tanto, o acesso ao código-fonte é um pré-requisito;
- A liberdade do usuário em redistribuir cópias de modo que ele possa ajudar outros (liberdade 2).
- A liberdade do usuário em distribuir cópias de versões por ele modificadas aos outros (liberdade 3). Desta forma, ele confere o direito a toda comunidade de beneficiar-se de suas mudanças. Para tanto, acesso ao código-fonte é um pré-requisito.

A crescente popularidade do Projeto GNU — que atraia cada vez mais o interesse da comunidade de programadores — motivou Stallman a estabelecer, em outubro de 1985, a Free Software Foundation (FSF), organização sem fins lucrativos com o objetivo de coordenar o desenvolvimento do projeto, bem como promover os ideais do *software* livre na sociedade (WILLIAMS, 2010, p. 105). O grande feito da iniciativa criada por Stallman, todavia, foi a criação da Licença Pública Geral GNU (*GNU General Public License* — GNU GPL) em 1989 a fim de licenciar os *softwares* produzidos pelo Projeto GNU de forma livre a partir das liberdades fundamentais conferidas aos seus usuários e refletindo os ideais do projeto. Como o próprio autor afirma, o código dos *softwares* que compõem o GNU não foram licenciados sob domínio público (STALLMAN, 1985, p. 30). No sistema legal da propriedade intelectual norte-americano, o detentor dos direitos autorais de uma obra é soberano para destiná-la a quaisquer fins e, valendo-se dessa característica legal, a GPL fora projetada de forma a subverter a premissa tradicional da função dos direitos autorais em restringir a circulação do bem intelectual a fim de garantir a sua exploração exclusiva; para isso, o texto da licença afirma que, no exercício desse direito, o detentor dele autoriza o livre uso, modificação e cópia do *software* em sua forma original ou modificada, sob a condição dessa nova cópia ou versão manter essas mesmas liberdades originais através da adoção compulsória da GPL (CARLOTTO; ORTELLADO, 2011, p. 84-85).

> O movimento do *software* livre é produto da subversão das tradicionais ideias de propriedade com relação aos "bens intelectuais". Originou-se da insatisfação relativa ao regime tradicional de direito autoral quando aplicado ao *software*, na medida em que ele impedia as possibilidades de se atender a objetivos que fossem além daqueles puramente econômicos. Nesse sentido, o movimento do *software* livre teve como escopo transformar a proteção da propriedade intelectual para criar bens intelectuais abertos, amplamente acessíveis tanto com relação ao uso, quanto com relação à possibilidade de inovação e modificação, não só do ponto de vista econômico, como também do ponto de vista cognitivo (LEMOS, 2005, p. 71-72).

O efeito legal da licença faz com que todo *software* derivado ou que inclui o código-fonte de outro *software* licenciado sob a GPL — ainda que em partes — tenha que ser obrigatoriamente distribuído sob a mesma licença, tornando as garantias da licença "virais" para demais obras (CARLOTTO; ORTELLADO, 2011, p. 84-85). Esse artifício legal é conhecido como *copyleft*, em oposição à definição de *copyright* tradicionalmente existente. A partir da perspectiva do *copyleft*, a violação de direitos autorais ocorre quando algum agente tenta transformar um *software* sob esse regime para o regime de *copyright*, tentando "fechar" o código-fonte, impedir o acesso a ele, impedir a sua livre redistribuição e ações similares (LEMOS, 2005, p. 72).

Um dos casos de "fechamento" ou "apropriação" de código-fonte notados por Stallman foi a implementação do editor de texto Emacs desenvolvida por James Gosling — que posteriormente, trabalhando na Sun Microsystems, viria a conceber a linguagem de programação Java durante a década de 1990. O Emacs havia começado a ser desenvolvido em Lisp no AI Lab em 1976 por Stallman e colaboradores (WILLIAMS, 2010, p. 82-86) para computadores PDP-10 executando o ITS. Em 1981, Gosling, doutorando na Universidade Carnegie Mellon, havia implementado uma versão do *software*, escrita em C, para ser portada para o sistema operacional Unix. Embora Gosling tenha inicialmente licenciado a sua implementação de forma informal a colaboradores buscando obter continuidade para o projeto, ele posteriormente vendeu seus direitos autorais sobre o *software* para a empresa UniPress, que, sob interesses financeiros, passou a restringir a distribuição de versões modificadas pela comunidade — dentre as quais as versões iniciais do GNU Emacs produzido pelo próprio Stallman, que gradualmente substituiu o código-fonte escrito por Gosling a fim de torná-lo livre (CARLOTTO; ORTELLADO, 2011, p. 84; WILLIAMS, 2010, p. 103-104).

A GPL desempenhou um papel fundamental no estabelecimento da comunidade de *software* livre que nasceu ao entorno do Projeto GNU, uma vez que ela resguardava a comunidade de ter seu esforço e trabalho — materializado através do código-fonte — apropriados e fragmentados por indivíduos ou empresas interessadas em lucrar através da apropriação de capital através da proteção à propriedade intelectual — o que havia ocorrido com o sistema BSD e o Emacs de Gosling (CARLOTTO; ORTELLADO, 2011, p. 84).

2.3.4 Open Source Initiative

O interesse público pelo *software* livre gradativamente aumentou no decorrer da década de 1990, sobretudo motivado pelo desenvolvimento do *kernel* Linux e a sua popularidade. Como apontam Carlotto e Ortellado (2011, p. 87), o primeiro precedente significativo de interesse da indústria no *software* livre veio através da decisão da Netscape em liberar o código-fonte de seu navegador — o Netscape Navigator (posteriormente Netscape Communicator) —através de uma licença livre. No verão do hemisfério norte, em 1995, o navegador da Netscape liderava o mercado de navegadores *web* com uma fatia de 80% do mercado sob um contexto de forte expansão e popularização da Internet (OSHRI; DE VRIES, 2008, p. 68). Interessada em participar do mercado, em meados de 1995 a Microsoft tentou impor a Netscape uma divisão do mercado, onde a última o suporte de seu navegador ao sistema operacional Windows; caso contrário, a Microsoft entraria no mercado com um navegador concorrente ao Netscape (OSHRI; DE VRIES, 2008, p. 68). Com a Netscape tendo recusado o ultimato, a Microsoft prosseguiu para a introdução de seu navegador no mercado, nomeado Internet Explorer, em agosto de 1995. O Internet Explorer havia sido baseado numa base de código-fonte licenciada pela Microsoft da *startup* Spyglass, Inc.; o anseio da Microsoft em obter a participação da Netscape no mercado a motivou a reescrever praticamente todo o código-fonte licenciado e a implementar rápidos ciclos de desenvolvimento para o produto em curtos períodos de tempo (OSHRI; DE VRIES, 2008, p. 68). Os investimentos da Microsoft deram resultado, e por volta do fim de 1997 a participação de mercado do Netscape havia sido reduzida a 50%, fazendo, no início do ano seguinte, a empresa operar com prejuízos, cortar seu programa de desenvolvimento e realizar demissões em seu quadro de funcionários (OSHRI; DE VRIES, 2008, p. 74).

Para a Netscape, o domínio da Microsoft no mercado de navegadores *web* permitiria a ela introduzir tecnologias e padrões proprietários em como as páginas *web* são servidas — o que abrange renderização gráfica, interação com o usuário e recursos similares — no lado do cliente. Naturalmente, o domínio no lado do cliente permitiria à empresa dominar o mercado de servidores *web*, uma vez que havia a possibilidade legal de apenas os servidores proprietários da Microsoft poderem implementar os recursos igualmente proprietários para servir as páginas *web* (CARLOTTO; ORTELLADO, 2011, p. 87). Diante desse cenário, a Netscape oficializou em 22 de janeiro de 1998 a sua decisão de liberar o código-fonte de seu

navegador sob uma licença livre baseada na GNU GPL e concentrar seus esforços no mercado de servidores, que lhe era mais rentável (CARLOTTO; ORTELLADO, 2011, p. 87).

Quittner e Slatalla (1999, p. 299-303), bem como Oshri e de Vries (2008, p. 75), citam que a decisão da Netscape em abrir o código-fonte do seu navegador foi influenciada por um livro branco de 20 páginas publicado por Frank Hecker, um de seus engenheiros. Hecker inspirou seu trabalho a partir da observação do aparente sucesso das comunidades de *software* de código aberto que já repercutiam entre a indústria e pelo artigo *The Cathedral and the Bazaar* escrito por Eric Steven Raymond — sugerido por Jamie Zawinski, um de seus colegas de trabalho.

> [...] é um memorando que eu escrevi argumentando para a Netscape abrir o código-fonte do seu *software*. É citado como tendo influenciado a decisão da Netscape em lançar o Netscape Communicator como um *software* de código aberto [...] embora eu tenha que ressaltar que minha proposta era, na verdade, licenciar comercialmente o código-fonte sob condições brandas, e distribuí-lo sem custos para uso não comercial — algo análogo a forma com a qual a Netscape licenciou as distribuições binárias do Netscape Navigator e do Netscape Communicator (HECKER, 2024, n.p, tradução nossa[3]).

O processo de abertura do código-fonte do navegador foi realizado por um time interno da Netscape. Raymond foi convidado a participar de uma reunião, realizada em 4 de fevereiro de 1998, com alguns dos executivos e técnicos da empresa, a fim de definir a estratégia de abertura e o licenciamento do código-fonte. Em 23 de fevereiro, a empresa criou a Mozilla Organization para facilitar a abertura do código-fonte, bem como atuar como um agente intermediário entre as contribuições para o código internas da Netscape e externas a ela. Em 31 de março, o código-fonte do Netscape Communicator foi finalmente publicado através do mozilla.org — o *website* da nova organização (OSHRI; DE VRIES, 2008, p. 75).

O sucesso do Linux, a articulação de Raymond em defesa do modelo de desenvolvimento aberto para *software* e a decisão da Netscape ajudou a atrair uma parcela da indústria receptiva ao *software* de código aberto, cujo modelo de desenvolvimento passou a ser percebido como mais eficiente e confiável; por sua vez, isso inspirou uma parcela dentro da comunidade do *software* livre a criar uma nova forma de se pensar e divulgar o modelo de código aberto — reformulando a filosofia do *software* livre a fim de torná-la mais atrativa aos interesses comerciais da indústria e do mercado; isso envolveu distanciar-se dos princípios

[3] No original: "[...] is an memo I wrote to argue for Netscape releasing the source code of its software. It's been cited as having influenced Netscape's decision to release Netscape Communicator as open source [...] although note that my proposal was actually to commercially license the source code under liberal terms, and provide it at no charge for noncommercial use — somewhat analogous to the way Netscape licensed the Netscape Navigator and Netscape Communicator binary distributions".

éticos e sociais defendidos pela Free Software Foundation e simpatizantes para enaltecer os benefícios ditos "práticos" ou "pragmáticos" associados ao código aberto e ao modelo transparente e público de desenvolvimento de *software* (O'MAHONY, 2007, p.140-141; RAYMOND, 1999, p. 96-101).

Em 3 de fevereiro de 1998, numa reunião realizada nos escritórios da VA Research em Mountain View, Califórnia, um grupo de desenvolvedores — liderados por Bruce Perens e Eric S. Raymond — decidiram fundar a Open Source Initiative ("iniciativa pelo código aberto" na língua portuguesa), uma organização sem fins lucrativos para promover o *software open source* — termo cunhado para diferenciar a nova visão proposta por eles daquela presente no *software* livre (RAYMOND, 1999, p. 96-101). Raymond argumenta que o novo termo foi necessário para dissociar a nova proposta da visão anterior, dado que o termo "*free*" na língua inglesa pode ser traduzido tanto como "livre" (no sentido de liberdade) quanto como "grátis" (no sentido de sem custos financeiros), apontando a ambiguidade como uma das causas que difundiam negativamente no mercado um preconceito do *software* de código aberto como o *software* incapaz de ser monetizado (RAYMOND, 1999, p. 98).

> Os dois termos descrevem quase a mesma categoria de *software*, porém eles referem-se a visões baseadas em valores fundamentalmente diferentes. Para o movimento do *software* livre, o *software* livre é um imperativo ético, com respeito essencial à liberdade dos usuários. Em contrapartida, a filosofia do código aberto considera os problemas em termos de como tornar o *software* "melhor" – e, um sentido prático apenas. Ela diz que o *software* não livre é uma solução inferior para o problema prático em questão (STALLMAN, 2015, p. 76, tradução da FSF[4]).

Para guiar a atuação do movimento e consolidar o seu entendimento sobre código aberto, a OSI adotou o *The Open Source Definition* (OSD) — um documento sob a forma de um contrato social derivado do *Debian Free Software Guidelines* ("diretrizes de *software* livre do Debian", pertencentes a distribuição Debian do GNU/Linux) contendo a sua definição de *software* de código aberto, bem como dez critérios para que um *software* possa ser caracterizado como tal (PERENS, 1999, p. 79; OSI, 2007, n.p):

- Livre redistribuição: a licença não deve restringir qualquer parte de vender ou doar o *software* como um componente de uma distribuição agregada de *software*

[4] No original: "the two terms describe almost the same category of software, but they stand for views based on fundamentally different values. Open source is a development methodology; free software is a social movement. For the free software movement, free software is an ethical imperative, essential respect for the users' freedom. By contrast, the philosophy of open source considers issues in terms of how to make software "better" — in a practical sense only. It says that nonfree software is an inferior solution to the practical problem at hand. Most discussion of "open source" pays no attention to right and wrong, only to popularity and success".

contendo programas de diversas fontes diferentes. A licença não deve exigir *royalties* ou outras taxas por tal venda;

- Código-fonte: o programa deve incluir seu código-fonte e deve permitir a sua distribuição em forma de código-fonte e também em sua forma compilada. Quando alguma forma do produto não for distribuída com seu código-fonte, deve haver um meio bem divulgado de se obter o código-fonte por não mais do que um custo de reprodução razoável, preferencialmente através de transferência gratuita pela Internet. O código-fonte deve ser a forma preferida na qual um programador modificaria o programa. Código-fonte deliberadamente ofuscado não é permitido. Formas intermediárias, como a saída de um pré-processador ou tradutor, não são permitidas;
- Obras derivadas: a licença deve permitir modificações e obras derivadas, e deve permitir que sejam distribuídos nos mesmos termos da licença do *software* original;
- Integridade do código do autor: a licença pode restringir a distribuição do código-fonte em sua forma modificada somente se a licença também permitir a distribuição de "arquivos de *patch*" com o código-fonte com a finalidade de modificar o programa em tempo de construção. A licença deve permitir explicitamente a distribuição de *software* construído a partir de código-fonte modificado. A licença pode exigir que trabalhos derivados tenham um nome ou número de versão diferente do *software* original;
- Não discriminação contra pessoas ou grupos: a licença não deve discriminar nenhuma pessoa ou grupo de pessoas;
- Não discriminação contra áreas de utilização: a licença não deve restringir ninguém de fazer o uso do programa em uma área de atuação específica. Por exemplo, não se deve restringir a utilização do programa numa empresa, ou ainda para pesquisa genética;
- Distribuição da licença: os direitos associados ao programa devem ser aplicados a todos a quem o programa for redistribuído sem a necessidade, para isso, da execução de uma licença adicional por essas partes;
- A licença não pode ser específica a um produto: os direitos associados ao programa não devem depender do programa fazer parte de uma distribuição de *software* específica. Se o programa for extraído dessa distribuição e usado ou distribuído dentro dos termos da licença do programa, todas as partes para quem o

programa for redistribuído deverão ter os mesmos direitos que aqueles concedidos em conjunto com a distribuição do *software* original;

- A licença não pode restringir outro *software*: a licença não deve impor restrições a outro *software* distribuído juntamente com o *software* licenciado. Por exemplo, a licença não deve exigir que todos os outros programas distribuídos no mesmo meio sejam *software* de código aberto;
- A licença deve ser neutra quanto a tecnologia: nenhuma disposição da licença pode ser baseada em qualquer tecnologia individual ou estilo de interface.

2.4 **Inovação**

O termo "inovação" se popularizou em Economia através do economista austríaco Joseph Alois Schumpeter em seu livro *Teoria do Desenvolvimento Econômico* (*Theorie der wirtschaftlichen Entwicklung* na língua alemã) publicado originalmente na língua alemã em 1911 e reeditado para uma versão na língua inglesa em 1934 sob o título *A Teoria do Desenvolvimento Econômico: Uma Investigação Sobre Lucros, Capital, Crédito, Juro e o Ciclo Econômico* (*The Theory of Economic Development: An Inquiry into Profits, Capital, Credit, Interest and the Business Cycle*). Um dos grandes destaques do trabalho produzido por Schumpeter foi a diferenciação entre invenção — fruto da pesquisa científica realizada pelo inventor — e a inovação — a comercialização da invenção no mercado realizada através do empreendedor ou empresário (SCHUMPETER, 1997, p. 95). Schumpeter diferencia invenção de inovação da seguinte forma:

> Uma invenção é uma ideia, esboço ou modelo para um novo ou melhorado artefato, produto, processo ou sistema. Uma inovação, no sentido econômico somente é completa quando há uma transação comercial envolvendo uma invenção e assim gerando riqueza (AGUSTINHO; GARCIA, 2018, p. 226, *apud* SCHUMPETER, 1988, p. 108).

Tidd, Bessant e Pavitt (2005, p. 68-69) aponta a dificuldade de definir o termo "inovação" com precisão devido a múltiplas maneiras de se entendê-lo, e ainda ressalta que é comum confundir o termo com invenção; todavia, os autores definem inovação como o processo de transformar oportunidades em novas ideias e também a prática de colocá-las em amplo uso. Chesbrough (2003), concordando com o entendimento promovido por

Schumpeter, entende por inovação a invenção implementada e levada ao mercado. Chesbrough introduz ainda o conceito de paradigma de inovação, entendido como um conjunto de práticas e formas de se pensar a produção e difusão de inovação em um determinado tempo. O autor analisa e identifica dois paradigmas gerais de inovação presentes nas sociedades capitalistas atuais: o paradigma da inovação fechada e o paradigma da inovação aberta.

2.4.1 Inovação fechada

O paradigma da inovação fechada predominou com significativo êxito nas economias capitalistas durante grande parte do século XX (CHESBROUGH, 2003a, p. 33-34). Nesse paradigma, o sucesso em inovar depende do controle total das ideias e recursos que a organização possui, exigindo então que ela gere, desenvolva, fabrique, comercialize, distribua, ofereça, financie e suporte seus produtos e serviços por conta própria. Desse modo, a organização assume uma postura de exclusiva auto confiança dado que ela não acredita na qualidade, capacidade e disponibilidade das ideias de terceiros. A propriedade intelectual surge como um meio da organização manter e apropriar-se — através de meios legais — das inovações desenvolvidas internamente através de sua infraestrutura de pesquisa e desenvolvimento (CHESBROUGH, 2003a, p. 33-34). Chesbrough aponta como uma das "regras implícitas" do paradigma da inovação fechada dos cinco seguintes pontos: i) contratação de profissionais de destaque na indústria; ii) descoberta e desenvolvimento de produtos e serviços por conta própria; iii) almejo por pioneirismo no mercado; iv) foco em liderar os investimentos em pesquisa e desenvolvimento na indústria; v) rígido controle da propriedade intelectual. Para o autor, tais fatores presentes na lógica da inovação fechada ocasiona em uma espécie de ciclo vicioso (Figura 1): as organizações investem em P&D interno, gerando muitas descobertas inovadoras que permite a elas comercializarem novos produtos e serviços para o mercado, o que por sua vez gera aumento de vendas e capital sendo que esse último é utilizado para reinvestir em P&D interno, gerando uma nova iteração de todo o processo. O controle rigoroso da propriedade intelectual possibilita que as organizações tenham o direito exclusivo de explorar as inovações, excluindo terceiros de fazer o mesmo e ter retorno financeiro com isso (CHESBROUGH, 2003a, p. 34).

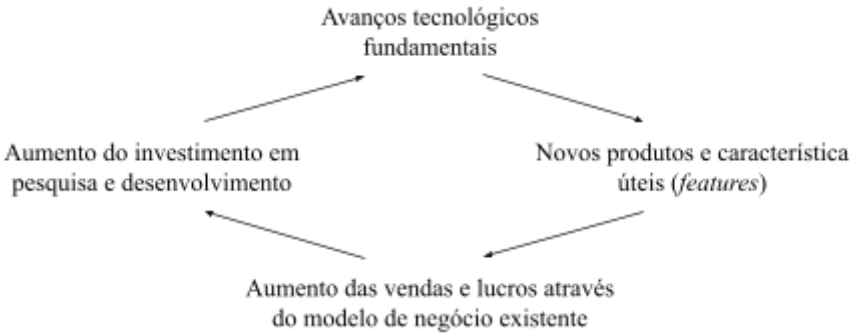

Figura 1 – Ciclo vicioso pertinente ao modelo de inovação fechada
Fonte: CHESBROUGH, 2003a, p. 34.

Historicamente, através de pesquisa e desenvolvimento interno, companhias alemãs da indústria química puderam sistematicamente expandir a oferta de produtos a partir das crescentes e avançadas investigações das propriedades dos materiais que elas estavam utilizando para criar novos corantes. Companhias da indústria petrolífera rapidamente aprimoraram suas capacidades em refinar o petróleo através do melhor entendimento das propriedades daquela matéria-prima, inovando ao criar novos produtos derivados dela (CHESBROUGH, 2003b, p. 24). O grande desempenho dessas instalações de P&D possibilitaram a criação de muitos monopólios naturais em diversas indústrias, desempenhando um papel crítico no surgimento das corporações industriais modernas. Essas instalações também se tornaram estrategicamente essenciais para as empresas e o investimento nelas passou a ser crítico para seus negócios (CHESBROUGH, 2003b, p. 24).

Analisando o contexto norte-americano, Chesbrough (2003b, p. 25) atribui como um dos fatores-chave para essa configuração o relacionamento entre as universidades públicas e as corporações privadas que surgiu no início do século XX. Em contraste com o sistema de ensino superior europeu, o sistema norte-americano era altamente descentralizado; as universidades públicas eram financiadas pelos estados e respondiam majoritariamente aos interesses comerciais locais. O efeito disso foi a formação de grandes quantidades de engenheiros e cientistas qualificados dos quais as corporações poderiam empregar em suas próprias instalações internas de pesquisa e desenvolvimento. Não o bastante, a participação dos Estados Unidos na Segunda Guerra Mundial catalisou a eficiência, produção e inovação na indústria norte-americana; ao final da guerra os avanços militares e científicos produzidos durante o conflito foram influentes para definir as políticas de promoção da ciência e tecnologia norte-americanas do pós-guerra. Até 1985, a maior parte do financiamento em

pesquisa e desenvolvimento no país era proveniente do governo. (CHESBROUGH, 2003b, p. 25-26).

Durante o pós-guerra, o governo norte-americano era o principal financiador de pesquisas no país. Todavia, a maior parte do trabalho de pesquisa não era realizado em instalações governamentais dedicadas, mas sim através de universidades públicas, operando através de investigações científicas cujos resultados eram livremente disseminados para a sociedade. A capacitação profissional gerada através dessa configuração possibilitou a expansão de muitos laboratórios privados formados antes da guerra, como o Bell Laboratories e os laboratórios da General Electric e DuPont; bem como levou à formação de novos laboratórios como o T. J. Watson Laboratories da IBM, o Sarnoff Labs da RCA, o HP Labs da Hewlett-Packard e o Xerox PARC da Xerox (CHESBROUGH, 2003b, p. 28). Esse período é descrito como o auge ou "a era de ouro" da inovação fechada, com o investimento privado sendo a "ponta de lança" da pesquisa científica. As maiores companhias da indústria conseguiam realizar e tirar maior proveito das inovações tecnológicas, alcançando as posições mais confortáveis no mercado. Essa configuração naturalmente criou uma barreira para possíveis competidores, uma vez que quaisquer novos competidores teriam que realizar investimentos nas mesmas ordens de grandeza para serem capazes de competir de forma justa no mercado (CHESBROUGH, 2003b, p. 29).

Nesse cenário, os objetivos distintos dos setores de pesquisa e setores de desenvolvimento corporativos possibilitaram a criação de "*buffers*", "silos", "bancos" ou ainda "armazéns" de ideias dentro das organizações. Isso porque nem toda invenção ou descoberta fruto do trabalho dos setores de pesquisa estava apta a ser prontamente desenvolvida e comercializada através de produtos ou serviços no mercado como uma inovação, tarefa essa a cargo dos setores de desenvolvimento. Com isso, as descobertas inaptas a serem desenvolvidas seriam mantidas (estocadas) em segredo nos "*buffers*", "bancos" ou "armazéns" de ideias da empresa até que ela julgasse ser o momento ideal para desenvolvê-las (CHESBROUGH, 2003b, p. 31-33).

Chesbrough (2003b, p. 34-40), aponta quatro principais fatores que causaram a erosão desse paradigma em diversas indústrias norte-americanas particularmente nas décadas finais do século XX; o primeiro fator foi o aumento da disponibilidade e mobilidade de profissionais qualificados no mercado de trabalho, que possibilitou com que o conhecimento armazenado internamente nas empresas fosse difundindo entre clientes, fornecedores, parceiros comerciais, universidades, *startups* e similares. Através da contratação de profissionais experientes e previamente treinados por uma empresa para seu quadro de funcionários, uma

empresa concorrente poderia beneficiar-se do conhecimento da primeira sem realizar grandes investimentos financeiros. Chesbrough exemplifica esse cenário através de uma análise histórica da indústria de discos rígidos norte-americana: funcionários do setor de discos rígidos da IBM colaboraram no estabelecimento de 21 das 99 empresas do setor entre 1973 e 1996 (CHESBROUGH, 2003b, p. 35). Estudos empíricos e teóricos na literatura organizacional acadêmica evidenciam esse fator desde a década de 1960 (KAISER; KONGSTED; RØNDE, 2015, p. 91-93; MØEN, 2005, p. 82).

O segundo fator foi o aumento da disponibilidade de capital de risco no mercado, possibilitando o surgimento de *startups* proeminentes; entre meados da década de 1970 e o ano 2000 evidenciou-se nos Estados Unidos um expressivo crescimento na angariação de fundos para capital de risco no país, de cerca de US$ 700 milhões em 1980 para mais de US$ 80 bilhões em 2000 (CHESBROUGH, 2003b, p. 38; GOMPERS; LERNER, 2003, p. 4). Essas *startups* representavam uma ameaça para os negócios das empresas já estabelecidas no mercado, uma vez que a ideia de altos retornos financeiros através de *startups* atraía funcionários dessas empresas para novos empreendimentos em potenciais. Este segundo fator relaciona-se intimamente com o primeiro, uma vez que a mobilidade de funcionários contribui para a difusão do conhecimento armazenado em empresas bem estabelecidas para as demais presentes na indústria (CHESBROUGH, 2003b, p. 37-38).

O terceiro fator está relacionado ao rápido ciclo de vida de produtos e serviços no mercado. Rápidas mudanças e avanços tecnológicos aliados à competitividade acirrada nas indústrias encurtou o ciclo de vida de produtos e serviços, isto é, o período entre a sua concepção e o seu declínio no mercado (GAIMON; SINGHAL, 1992, p. 211-214; VESEY, 1992, p. 151-152). Esse fator tornou-se mais perceptível nas indústrias com mais dependência de novas tecnologias como a automobilística e eletroeletrônica (VESEY, 1992, p. 151-152). A crescente demanda por produtos e serviços melhores e a competição acirrada com novas *startups* no mercado forçou as empresas altamente dependentes de pesquisa e desenvolvimento interno a aumentar o ritmo com o qual elas processavam ideias e inovações em produtos comercializáveis (*time to market*); isso significou que tais empresas já não mais poderiam armazenar ideias ou inovações internamente, uma vez que elas poderiam facilmente "vazar" para o ambiente externo da empresa devido a alta mobilidade de funcionários, oferecendo margem para que competidores prontamente se apropriarem dessas inovações; tão logo quanto uma invenção fosse descoberta nas instalações de pesquisa, ela teria que ser desenvolvida, sob a forma de uma inovação, em um produto para o mercado (CHESBROUGH, 2003b, p. 38-39). Gaimon e Singhal (1992, p. 211) exemplificam esse fator

através da atuação da IBM na indústria de computadores *mainframes*: no período de 12 anos entre 1964 e 1976, a IBM introduziu no mercado duas famílias de *mainframes* — a System/360 e System/370; todavia, nos quatro anos subsequentes ao período citado a empresa introduziu quatro novas famílias de *mainframes* — agrupadas nas linhas IBM 303X e IBM 308X.

O quarto e último fator, de acordo com Chesbrough, trata-se do aumento da competência de fornecedores externos no mercado. A expansão das universidades e do ensino superior, o aumento da disponibilidade de profissionais capacitados e capital de risco no mercado tornou fornecedores externos mais capacitados e competitivos, uma vez que eles passaram a ser capazes de ofertarem produtos ou serviços de qualidade igual ou superior aos das empresas bem estabelecidas no mercado (CHESBROUGH, 2003b, p. 39-40). Chesbrough aponta ainda que o aumento da competência de fornecedores externos possibilitou com que as empresas bem posicionadas no mercado se expandissem mais rapidamente, pois elas não mais precisariam realizar todos os processos de suas cadeias de valor por conta própria.

Os fatores que provocaram a erosão do paradigma da inovação fechada alteraram significativamente a distribuição do conhecimento na sociedade. Se antes o conhecimento estava confinado aos grandes centros corporativos de P&D, na atualidade ele se encontra distribuído em "poças" na sociedade, como em usuários, fornecedores, universidades, laboratórios nacionais, consórcios, consultorias e *startups* (CHESBROUGH, 2003b, p. 40).

2.4.2 Inovação aberta

O termo inovação aberta ("*open innovation*" na língua inglesa) foi cunhado por Henry William Chesbrough em 2003 para definir um novo e perceptível fenômeno organizacional caracterizado pela abertura do processo de inovação de empresas privadas. Chesbrough, Vanhaverbeke e West definem a inovação aberta como sendo:

> [...] o uso dos fluxos de entrada e saída de conhecimento para acelerar a inovação interna e expandir os mercados para o uso externo da inovação, respectivamente. A inovação aberta é um paradigma que pressupõe que as empresas podem e devem utilizar ideias externas, bem como ideias internas, e caminhos internos e externos

> para o mercado, à medida em que procuram avançar com sua tecnologia (CHESBROUGH; VANHAVERBEKE; WEST, 2006, p. 2, tradução nossa[5]).

Chesbrough ainda complementa o conceito da seguinte maneira:

> [...] a inovação aberta combina ideias internas e externas em arquiteturas e sistemas cujos requisitos são definidos por um modelo de negócio. O modelo de negócio utiliza ambas as ideias internas e externas para criar valor, enquanto define mecanismos internos para reclamar parte desse valor. A inovação aberta pressupõe que as ideias internas também podem ser levadas ao mercado por meio de canais externos, fora do negócio atual da empresa, para criar valor adicional (CHESBROUGH, 2003a, p. 37, tradução nossa[6]).

Enquanto que os trabalhos acadêmicos iniciais acerca da inovação aberta abordaram primariamente o processo de pesquisa e desenvolvimento, trabalhos posteriores buscaram a partir disso expandir suas perspectivas de análise, levando a abertura de novas linhas de pesquisa. Gassmann, Enkel e Chesbrough (2010, p. 213-214) agrupam as múltiplas linhas de pesquisa acerca da inovação aberta em nove perspectivas distintas (Quadro 1).

Quadro 1 – As perspectivas da pesquisa acadêmica em inovação aberta e seus respectivos objetivos
(continua)

Perspectiva	Objetivos
Espacial	Busca estudar aspectos relativos à inovação e a inovação aberta considerando o mundo globalizado, sobretudo através das novas tecnologias da informação e comunicação.
Estrutural	Busca estudar aspectos relativos à inovação aberta na estruturação das indústrias diante de cadeias de valor cada vez mais desagregadas.
Centrada no usuário	Busca estudar o papel dos usuários dentro dos processos de inovação das organizações, bem como seus papéis na concepção e democratização da inovação.
Centrada no fornecedor	Busca compreender a integração dos fornecedores dentro dos processos de inovação aberta das organizações e seus efeitos.
Potencial	Busca entender as potenciais formas com as quais a inovação aberta é e pode vir a ser comercializada através de diferentes modelos de negócios.

[5] No original: "[...] the use of purposive inflows and outflows of knowledge to accelerate internal innovation, and expand the markets for external use of innovation, respectively. Open innovation is a paradigm that assumes that firms can and should use external ideas as well as internal ideas, and internal and external paths to market, as they look to advance their technology".

[6] No original: "[...] open Innovation combines internal and external ideas into architectures and systems whose requirements are defined by a business model. The business model utilizes both external and internal ideas to create value, while defining internal mechanisms to claim some portion of that value. Open Innovation assumes that internal ideas can also be taken to market through external channels, outside the current businesses of the firm, to generate additional value".

(conclusão)

Perspectiva	Objetivos
Centrada no processo	Busca estudar as práticas ou processos com os quais as empresas podem importar e exportar o conhecimento a partir da abertura de sua cadeia de inovação; tais processos podem ser tipificados em "de fora para dentro" (*inbound*), "de dentro para fora" (*outbound*) ou ainda "acoplados" (*coupled*).
Ferramental	Busca compreender os instrumentos e mecanismos com os quais as empresas podem abrir o seu processo de inovação.
Institucional	Busca estudar a inovação aberta como um modelo de inovação simultaneamente privado e coletivo através das teorias acadêmicas para inovação.
Cultural	Busca entender a inovação aberta como fruto de uma mudança na mentalidade dos agentes que dela participam ocasionada por fatores como valores empresariais, sistemas de incentivos, plataformas de comunicação, critérios de tomada de decisão, dentre outros.

Fonte: Adaptado de Gassmann, Enkel e Chesbrough (2010, p. 213-214).

Chesbrough, Vanhaverbeke e West (2006, p. 286-287) afirmam que grande parte dos estudos sobre a inovação aberta concentram-se em analisar o paradigma a nível da empresa, devido a duas razões: a primeira delas é o entendimento tradicional da inovação como produto das ações de uma única empresa; a segunda é que o valor de uma inovação só pode ser notado através de um modelo de negócio de uma empresa. Todavia, os autores argumentam que tanto a prática quanto o estudo da inovação aberta não se limitam ao nível da empresa, uma vez que inovações são produtos de pessoas trabalhando de forma individual ou em grupo através de organizações; simultaneamente, empresas estão imersas em redes, indústrias e setores, num dado contexto político e econômico. Os autores propõem, portanto, cinco níveis de análise da inovação aberta: o indivíduo, a organização, a rede de valor, a indústria/setor e as instituições nacionais (CHESBROUGH; VANHAVERBEKE; WEST, 2006, p. 286-287).

Chesbrough, Vanhaverbeke e West (2006, p. 286) afirmam que a inovação aberta é simultaneamente um conjunto de práticas para a empresa beneficiar-se economicamente da inovação e também um modelo cognitivo para criar, interpretar, e estudar tais práticas. Desde o surgimento dos estudos acerca da inovação aberta, a literatura tem identificado um vasto conjunto de práticas pertencentes ao paradigma, entre as quais se destacam (CÂNDIDO, 2018, p. 147):

- A utilização de fontes externas de informação e conhecimento;
- A realização de atividades de pesquisa e desenvolvimento em cooperação com universidades, clientes, fornecedores e concorrentes;
- A formação de alianças estratégicas;

- A obtenção de financiamento externo para projetos inovadores;
- O licenciamento da propriedade intelectual da empresa;
- A observação das práticas de inovação realizadas em outras empresas;
- A aplicação da inovação entre as indústrias.

A efetividade das práticas promovidas pela inovação aberta depende, todavia, do sucesso da empresa em formular um modelo de negócio capaz de capturar valor a partir da inovação — uma parte crítica do paradigma da inovação aberta. A falha em alcançar esse objetivo pode fazer com que a inovação traga menos valor para a empresa que ela traria em um caso de sucesso; para além disso, concorrentes podem ter sucesso nesta tarefa e serem capazes de formular um modelo de negócio para a inovação que a empresa descobriu e obterem vantagem competitiva sobre ela (CHESBROUGH, 2003b, p. 64). De acordo com Chesbrough (2003b, p. 64) "uma tecnologia medíocre suportada por um bom modelo de negócio pode ser mais valiosa que uma boa tecnologia suportada por um modelo de negócio medíocre". Para Chesbrough (2003b, p. 64), são funções de um modelo de negócio:

- Articular uma proposição de valor, isto é, o valor criado para os usuários a partir da oferta da tecnologia;
- Identificar um segmento de mercado, isto é, os usuários para os quais a tecnologia será útil e consequentemente ofertada;
- Definir a estrutura da cadeia de valor da empresa, a qual é necessária a fim de criar e distribuir a oferta, bem como determinar os recursos complementares necessários para suportar a posição da empresa na cadeia;
- Especificar os mecanismos de geração de receita para a empresa, para além de estimar a estrutura de custos e a margem de receita desejada de produção da oferta de acordo com a proposição de valor e a estrutura da cadeia de valor selecionadas;
- Descrever a posição da empresa junto a rede de valor, ligando fornecedores e clientes, incluindo a identificação de potenciais empresas complementares e concorrentes;
- Formular a estratégia competitiva pela qual a empresa ganhará e manterá vantagem sobre as rivais.

Os processos (ENKEL; GASSMANN; CHESBROUGH, 2009, p. 312-313) ou atividades (CÂNDIDO, p. 149) referentes a inovação aberta podem ser agrupados, de acordo com Chesbrough e Crowther (2006, p. 229), em uma única dimensão que considera a existência de dois fluxos possíveis para transferência de conhecimento — *inbound* ("para dentro") e *outbound* ("para fora"). Enkel, Gassmann e Chesbrough (2009, p. 313) propõem

ainda um terceiro fluxo — *coupled* (“acoplada”). Esses fluxos são nomeadas de acordo com a forma pela qual o conhecimento flui nas interações realizadas pela empresa: no fluxo *inbound*, busca-se obter ideias ou conhecimentos externos para o ambiente interno da empresa; no fluxo *outbound*, de forma inversa, a empresa busca disponibilizar o seu conhecimento para o ambiente externo; consequentemente no fluxo *coupled*, há simultaneamente ambos os fluxos anteriores (ENKEL; GASSMANN; CHESBROUGH, 2009, p. 312-313). O Quadro 2 reúne os fluxos da inovação aberta e suas respectivas práticas comuns exemplificadas de acordo com Bigliardi e Galati (2016, p. 3).

Quadro 2 – Fluxos da inovação aberta e suas respectivas práticas comuns

Fluxo	Exemplo de prática	Descrição
Inbound	Participação de clientes	Envolver diretamente o cliente no processo de inovação (por exemplo, desenvolvendo produtos com base nas especificações fornecidas por clientes).
	Colaboração externa	Colaborar sistematicamente com parceiros externos (por exemplo, universidades, centros de pesquisa ou outras empresas) no processo de inovação.
	Incorporação de propriedade intelectual externa	Comprar ou utilizar propriedade intelectual externa (por exemplo, adquirindo patentes).
	Exploração da Internet	Utilizar sistematicamente a Internet como meio de exploração de ideias ou tecnologias inovadoras.
	Aquisição de conhecimento	Adquirindo o trabalho de pesquisa e desenvolvimento realizado por outros agentes.
Outbound	Comercialização de propriedade intelectual interna	Negociar licenças de patentes ou conhecimento desenvolvido internamente.
	Revelação de conhecimento	Disponibilizar gratuitamente o conhecimento interno não utilizado.
	Exportação de conhecimento	Participar ativamente no processo de inovação de outros agentes.
Coupled	Alianças com empresas complementares	Combinar os processos da dimensão *inbound* (para adquirir conhecimento) com os da dimensão *outbound* (para exportar ideias para o mercado).

Fonte: Adaptado de Bigliardi e Galati (2016, p. 3).

Trabalhos iniciais em inovação aberta como Chesbrough (2003b, p. 64) e Chesbrough, Vanhaverbeke e West (2006, p. 101) inicialmente defende o papel crítico da presença de um modelo de negócio capaz de capturar valor a partir da inovação — gerando então retorno financeiro imediato para a empresa. Chesbrough, Vanhaverbeke e West (2006, p. 101), de

maneira mais rigorosa, defendem que a presença de processos *inbound* ou *outbound* na ausência de um modelo de negócio para a inovação não caracteriza inovação aberta. Todavia, autores posteriores, como Dahlander e Gann (2010), propõem que tal mecanismo não é fundamentalmente necessário para a inovação aberta em si. Para defender essa ideia, Dahlander e Gann (2010, p. 699-709) apontam que há uma ambiguidade na literatura sobre o que de fato significa o conceito de "abertura" (*openness* na língua inglesa); Laursen e Salter (2006, p. 131-150), por exemplo, definem "abertura" como o grau em que a empresa é capaz de absorver conhecimento de fontes externas, composto pelas variáveis "abrangência" e "profundidade"; por outro lado, Henkel (2006, p. 956) define "abertura" como o grau em que a empresa é capaz de exteriorizar o conhecimento presente em seu ambiente interno, influenciado por fatores gerenciais como a proximidade no relacionamento da empresa com os agentes externos. A partir de uma análise da literatura empírica, Dahlander e Gann propõem, portanto, que "abertura" pode ser compreendida tanto como o grau em que a empresa se propõe a incorporar quanto exteriorizar conhecimento, influenciada por uma série de fatores, sendo a presença ou não de um modelo de negócio um deles. Portanto, para os autores, "abertura" não pode ser compreendida como uma variável binária, capaz de assumir somente os valores "aberto" e "fechado", nem tão pouco a ausência de um modelo de negócio é motivo o suficiente para caracterizar um processo de inovação como fechado (DAHLANDER; GANN, p. 702-703). Tendo isso em mente, Dahlander e Gann propõem um modelo que adiciona uma dimensão composta por processos pecuniários (onde há a comercialização da inovação) e não pecuniários (onde não há a comercialização da inovação) a dimensão inicial *inbound* e *outbound*. A partir disso, os processos de inovação aberta podem ser enquadrados em quatro tipos genéricos: i) aquisição (*acquiring*) para processos de inovação aberta *inbound* pecuniários; ii) obtenção (*sourcing*) para processos de inovação aberta *inbound* não pecuniários; iii) comercialização (*selling*) para processos de inovação aberta *outbound* pecuniários; e por fim iv) revelação (*revealing*) para processos de inovação aberta *outbound* não pecuniários. O Quadro 3 descreve a matriz composta pelo modelo de duas dimensões proposto.

Quadro 3 – Matriz da tipificação de processos conforme o modelo de dimensões da inovação proposto por Dahlander e Gann (2010)

	Quanto ao fluxo do conhecimento	
Quanto à comercialização	*Inbound*	*Outbound*
Pecuniários	Aquisição	Comercialização
Não pecuniários	Obtenção	Revelação

Fonte: Adaptado de Dahlander e Gann (2010, p. 702).

A aquisição refere-se aos processos de inovação aberta *inbound* onde a empresa utiliza o conhecimento externo sob a posse de outros agentes no mercado para complementar o seu processo de inovação interno. Aqui, há a presença de alguma espécie de aquisição financeira por parte da empresa, como por exemplo, através do licenciamento de propriedade intelectual de outras empresas. As competências internas da empresa em identificar e avaliar qual conhecimento é necessário adquirir é um fator crítico para este processo, dado que aquele conhecimento externo muito distante do domínio da empresa traz dificuldades no momento em que é incorporado ao dela, assim como conhecimento externo muito próximo dificulta a formação de novas combinações de ideias inovadoras (DAHLANDER; GANN, 2010, p. 705).

A obtenção refere-se aos processos de inovação aberta *inbound* onde a empresa absorve o conhecimento externo livre para complementar o seu processo de inovação interno. Diferentemente da aquisição, a obtenção não envolve transações comerciais, assumindo também que o conhecimento externo está distribuído livremente em inúmeras fontes em vez de tão somente sob a posse de agentes bem definidos no mercado. A capacidade da empresa em absorver conhecimento está intimamente relacionada ao nível em que ela confia nas ideias presentes no ambiente externo. Todavia, Dahlander e Gann apontam que pesquisas empíricas demonstram que o emprego extensivo da obtenção no processo de inovação pode comprometer sua performance, uma vez que há o risco de se consumir um tempo em pesquisa excessivamente grande, bem como o risco da empresa tornar-se excessivamente dependente de fontes externas para inovação (DAHLANDER; GANN, 2010, p. 704-705).

A comercialização refere-se aos processos de inovação aberta *outbound* onde a empresa exterioriza seu conhecimento interno através de transações comerciais. Na comercialização, a empresa vende ou licencia suas invenções ou tecnologia para o mercado, gerando a partir disso retorno financeiro para si. A empresa deve ser capaz, portanto, de precisar o que ela deve ou não comercializar, identificar ideias similares já presentes no

mercado, bem como antecipar o potencial valor do conhecimento a ser ofertado através de um planejamento estratégico (DAHLANDER; GANN, 2010, p. 704).

Em contraste com a comercialização, a "revelação" refere-se aos processos de inovação aberta *outbound* onde a empresa exterioriza seu conhecimento interno sem ganhos financeiros imediatos, almejando outros benefícios de forma indireta. A capacidade em se obter lucro a partir da inovação é governada diretamente pelo regime de apropriabilidade no qual a empresa se insere. As dimensões mais importantes desse regime são a natureza da inovação (envolvendo o seu produto, processos, bem como se ela é fruto de conhecimento tácito ou codificado) e a eficácia dos mecanismos formais/legais para proteção à propriedade intelectual (TEECE, 1986, p. 287). Para além de mecanismos legais, empresas utilizam ainda mecanismos ditos informais para obterem maior apropriabilidade de suas inovações, como empregar curtos tempos de ciclo (*lead time*) a fim de por sua inovação no mercado mais rapidamente (ou seja, em menor *time to market*), almejar pioneirismo no mercado ou aprisionar sua base de clientes a sua inovação (DAHLANDER; GANN, 2010, p. 703; SMIT, 2014, p. 3). Empresas que escolhem abrir seu processo de inovação através do emprego da revelação operam sobre regimes de apropriabilidade categorizados como "fracos" por Teece (1986, p. 287), onde a inovação é facilmente replicada por concorrentes no mercado. Dahlander e Gann (2010, p. 703) apontam, portanto, que empresas em tal posição optam por favorecer o emprego de mecanismos informais para apropriação do valor da inovação em desfavor de mecanismos formais como proteção à propriedade intelectual visando maior colaboração e resultados efetivos a curto prazo.

O paradigma da inovação aberta assume que o dinamismo do mercado exige que empresas não mais estoquem suas tecnologias até que elas julguem estarem prontas para serem comercializadas, uma vez que isso incorre no risco da empresa perdê-las para outros competidores (CHESBROUGH, 2003b, p. 41). Atualmente, não se encontra no mercado negócios exclusivamente baseados nas práticas promovidas pela inovação aberta, mas sim um misto delas com as práticas tradicionais provenientes do paradigma da inovação fechada. Abertura em larga escala pode impactar negativamente a longo prazo o sucesso da empresa em inovar, uma vez que pode ocasionar na perda e descontrole das competências-chave da empresa; da mesma forma, como já demonstrado, o paradigma da inovação fechada não atende a crescente demanda por curtos ciclos de inovação e reduzido *time to market*. Deve-se então almejar um balanceamento entre os dois paradigmas, onde a empresa busca utilizar todas as ferramentas ao seu alcance para criar e entregar produtos e serviços mais rapidamente que seus concorrentes ao mesmo tempo em que busca solidificar suas competências-chave e

proteger a sua propriedade intelectual (ENKEL; GASSMANN; CHESBROUGH, 2009, p. 312).

2.5 **Inovação aberta através de software de código aberto**

O processo de concepção, desenvolvimento e distribuição de *software* tem evoluído e se alterado com o passar das décadas. Situada no moderno contexto econômico caracterizado por acirrada competição decorrente da desregulação, consumidores empoderados, tecnologias emergentes, economia globalizada, incertezas quanto ao futuro econômico, curto ciclo de vida de tecnologias e o rápido desenvolvimento de novos produtos e serviços (EDISON; BIN ALI; TORKAR, 2013, p. 1390), a indústria de *software* torna-se uma das mais afetadas por tais mudanças dada a sua natureza de ser composta pelo uso intensivo de conhecimento e avançar conforme o surgimento de novas tecnologias (ROMIJN; ALBALADEJO, 2002, p. 1053-1067).

Para adaptar-se ao novo contexto, as empresas de *software* adotaram novas práticas com o intuito de acelerar seus ciclos de inovação quanto às mudanças do mercado. Modelos de desenvolvimento de *software* tradicionais, como o modelo em cascata, formulados a partir do final da década de 1960 para contornar a "crise do *software*" vigente até então, deram lugar no início da década de 1990 às chamadas metodologias ágeis (SOARES, 2004, p. 2; JIANG; EBERLEIN, 2009, p. 3735). Enquanto as metodologias tradicionais alcançaram ampla adoção e sucesso inicial, elas rapidamente perderam efetividade no final do século XX devido a característica de assumirem um longo ciclo de desenvolvimento, bem como estabilidade em requisitos, tecnologias e mercado, o que já não era mais uma realidade, em especial para produtos destinados a *web* (JIANG; EBERLEIN, 2009, p. 3735-3736). As metodologias ágeis, portanto, foram desenvolvidas a fim de tornar o processo de desenvolvimento de *software* mais incremental, flexível e reativo às mudanças, assumidas como certas quanto às suas ocorrências e imprevisíveis quanto ao seu conteúdo (JIANG; EBERLEIN, 2009, p. 3733-3734; DZAMASHVILI FOGELSTRÖM *et al*., 2010, p. 52-56).

Similarmente, a indústria se deparou com o desafio centrado na questão em como sustentar o *software* empacotado, predominante no mercado desde a década de 1970, em regimes de apropriabilidade de valor da inovação fortemente dependentes da proteção à propriedade intelectual em um ambiente marcado por cada vez mais mudanças tecnológicas.

Neste cenário, o curto ciclo de desenvolvimento do *software* e novas tecnologias de informação, usualmente medido em meses, torna-se um problema para empresas que consideram operar num regime de apropriabilidade fortemente dependente de proteção à propriedade intelectual, uma vez que os processos legais para registro podem levar, em média, anos para ser concluído (COCKBURN; MACGARVIE, 2011, p. 920); desta maneira, evidencia-se que os mecanismos tradicionais para proteção à propriedade intelectual não conseguem acompanhar o ritmo com o qual a indústria de *software* evolui e se transforma — oferecendo num curto espaço de tempo uma novas soluções tecnológicas e tornando obsoletas as anteriores (MATOS; TAVARES; FERREIRA, 2011, p. 4).

> Assim, o desenvolvimento de mecanismos de apropriabilidade na área de *software* não se esgota na aplicação do sistema de patentes, nem no registro de programas de computador, uma vez que as relações entre custo, estrutura de mercado e preço podem determinar as melhores estratégias de apropriabilidade. Uma gestão eficiente desses ativos que consiga auferir de modo efetivo resultados econômicos está condicionada à implantação de uma cultura de propriedade intelectual, de modo a articular esses ativos a outros intangíveis não passíveis de proteção, proporcionando um melhoramento dos produtos, agregação de valor aos mesmos e fomento à atividade inovativa (MATOS; TAVARES; FERREIRA, 2011, p. 4).

O progresso das tecnologias de redes de informação no início da década de 1990, através da convergência de três tendências — a digitalização da rede de telecomunicações, o desenvolvimento da transmissão em banda larga e a melhoria no desempenho de computadores conectados pela rede — moveu o paradigma da computação em rede, até então limitada meramente a "conexão" a redes locais, para a "cooperação" através de redes remotas, introduzindo o conceito de "computação cooperativa" (CASTELLS, 2002, p. 231). Esses avanços permitiram ainda o surgimento de processos flexíveis de gerenciamento, produção e distribuição de forma interativa através de computadores (CASTELLS, 2002, p. 231). O surgimento do *software* livre e de código aberto (do inglês *free and open-source software*, FOSS) na década de 1980 e sua continuidade na década seguinte trouxe um novo paradigma para a produção de *software* no ecossistema da computação, inicialmente com pessoas se unindo em comunidades para inovar e desenvolver *software* de maneira cooperativa e aberta sobre a pauta fortemente ideológica centrada nos direitos de utilização, modificação e distribuição de *software*. Na atualidade, observamos também uma colaboração crescente entre corporações da área de tecnologia da informação (TI) e comunidades de *software* de código aberto. Entre 2009 e 2019, o Linux, um dos mais influentes casos de sucesso de código aberto, recebeu continuamente contribuições anuais provenientes de 400 corporações para o seu repositório de código aberto (STEWART, KHAN e GERMAN, 2020, p. 13).

Atualmente, comunidades para o desenvolvimento de *software* de código aberto se formam virtualmente através da Internet, dada a facilidade de troca de dados e informações através da rede. Tipicamente, desenvolvedores voluntários contribuem para projetos de *software* de código aberto sem receber em troca nenhuma remuneração financeira, e inúmeros estudos disponíveis na literatura acadêmica do comportamento organizacional fornecem uma ampla gama de explicações para o fenômeno, incluindo motivações altruístas, desejo de aprendizado e troca de conhecimento ou diversão (GEROSA *et al*., 2021, p. 1046-1056; CHOI; PRUETT, 2015, p. 109-117). Predomina o desenvolvimento colaborativo e descentralizado de *software*, suportado pelo emprego de sistemas de controle de versões como o Concurrent Versions System (CVS), Apache Subversion (SVN), Mercurial e Git (ZOLKIFLI; NGAH; DERAMAN, 2018, p. 408-414). A criação do Git por Linus B. Torvalds em 2005 para dar suporte ao desenvolvimento do *kernel* Linux, popularizou o conceito de sistemas de controle de versões distribuídos, onde o código-fonte do *software* — disponível a partir de servidores na *web* — pode ser bifurcado em múltiplas ramificações as quais podem ainda ser desenvolvidas e modificadas de forma independente por qualquer desenvolvedor em qualquer parte do mundo para serem posteriormente fundidas ao código-fonte original — evoluindo assim o desenvolvimento do *software* — ou alternativamente serem publicadas como *softwares* derivados independentes (*forks*) (ZOLKIFLI; NGAH; DERAMAN, 2018, p. 408-414). Plataformas *web* como o SourceForge, Bitbucket, Launchpad, GitHub e GitLab fornecem ainda serviços gratuitos ou pagos para a criação de repositórios para hospedagem de código-fonte, bem como demais ferramentas e aplicações úteis para dar suporte ao desenvolvimento do *software* (SAFARI *et al*., 2020, p. 194-195).

No meio corporativo, entretanto, as reações iniciais ao *software* de código aberto nem sempre foram amistosas. Em meados da década de 1990, por exemplo, a Microsoft dominava o mercado de sistemas operacionais para uso doméstico com o Windows NT — seu principal produto — e investia para replicar o sucesso do sistema aumentando a sua participação no mercado de sistemas operacionais para uso corporativo, o apresentando como um forte concorrente para o NetWare, da Novell, e o OS/2 da IBM, bem como as inúmeras variantes do Unix presentes no mercado. Entre 1996 e 1997, as vendas do Windows NT no segmento cresceram oito vezes a taxa média de crescimento para todo o segmento (SOFTWARE PUBLISHERS ASSOCIATION, 1998, p. 32).

Neste cenário de competitividade, no final de outubro de 1998, uma série de memorandos internos confidenciais da Microsoft — posteriormente conhecidos como "documentos do *Halloween*" (*Halloween documents* na língua inglesa) — vieram a público

por meio do ativista do movimento de código aberto Eric S. Raymond, autor do influente ensaio *The Cathedral and the Bazaar*, que relata suas observações do processo desenvolvimento do *kernel* Linux (BIRKINBINE, 2020, p. 63). Os memorandos analisavam aspectos relativos ao *software* de código aberto — em especial o Linux —, identificando o movimento como uma verdadeira ameaça potencial aos negócios da empresa e trazendo potenciais estratégias para que ela se mantivesse competitiva nesse cenário, evidenciando também os impactos da disruptividade para com o mercado de *software* proprietário e relatando ainda os seguintes fatores (VALLOPPILLIL, 1998, p. 16-17; VALLOPPILLIL e COHEN, 1998, p. 14): i) o projeto possuía curtos períodos de interação — usualmente semanas — em seu processo de desenvolvimento, fornecendo novas funcionalidades e correções para o *software* em um tempo menor quando comparado com a doutrina de versões estáveis comuns no mercado até então; ii) o fato do sistema ser distribuído livremente a partir de servidores FTP (*File Transfer Protocol*) distribuídos pela rede impossibilitava estimar com precisão a sua adoção; iii) a liberdade do usuário em poder instalar o sistema em quantos dispositivos ele deseja-se sem nenhuma espécie de ônus financeiro representava uma clara desvantagem para a prática do mercado em impor limites na difusão do *software* através de mecanismos de licenciamento por cópia comercializada — como chaves de produto — ou limitação na quantidade de usuários para uma mesma negociação de cópia; Dirigindo-se a Microsoft, Valloppillil (1998, p. 7), cita ainda que “em outras palavras, para entender como competir com o *software* de código aberto, precisamos analisar o processo no lugar do produto”.

O posicionamento da Microsoft perante a comunidade de código aberto se modificou gradualmente ao longo da década de 2000; embora a empresa ainda visse *softwares* de código aberto como competidores as suas soluções de código fechado, surgiu nesta época um movimento gradual de colaboração com as comunidades de código aberto. Em 2006, o então diretor-executivo da empresa, Steve Ballmer, anunciou uma parceria com a Novell, que em 2003 havia adquirido a SuSE Linux AG, empresa alemã responsável pelo desenvolvimento fechado[7] da distribuição GNU/Linux SUSE Linux, que em 2005 teria seu desenvolvimento bifurcado — de maneira aberta — para a comunidade sob a forma da distribuição openSUSE (MARQUES e FILHO, 2008, p. 152). A parceria visava fomentar a colaboração técnica entre

[7] Embora pelos termos da licença GNU GPLv2 do *kernel* Linux a empresa fosse legalmente obrigada a disponibilizar o código-fonte do SuSE Linux, ela empregava um modelo de desenvolvimento fechado, pois o código-fonte com as modificações era desenvolvido internamente e não era prontamente disponibilizado ao público; somente clientes licenciados tinham acesso antecipado ao código do sistema operacional. Em contrapartida, o openSUSE empregou um modelo de desenvolvimento inteiramente aberto ao público.

as duas empresas em favor de suas respectivas bases de clientes, aprimorando a interoperabilidade entre os dois sistemas — Windows e SUSE Linux —, incluindo ainda um acordo de não reivindicação de patentes de eventuais tecnologias da Microsoft que pudessem ser incorporadas ao SUSE Linux que, sendo uma distribuição GNU/Linux, era consequentemente um projeto de código aberto (LAKKA, VAROUTAS e MARTAKOS, 2009). Em julho de 2009, em um importante marco, a Microsoft realizou a primeira contribuição direta para o *kernel* Linux, com a adição experimental de *drivers* para aprimorar a virtualização do sistema através da tecnologia de hipervisor Hyper-V proprietária da Microsoft, disponível nos sistemas operacionais Windows Server 2008 e Windows Server 2008 R2, a fim de atender a demanda de clientes corporativos que desejavam maior interoperabilidade entre os sistemas (SCHWARTZ, 2009; MICROSOFT, 2009). A década de 2010 que se seguiu foi marcada por uma profunda reestruturação das estratégias da empresa no mercado. Em um cenário de rápido avanço de novas tecnologias como a computação móvel, embarcada e baseada em nuvem, o Windows perde o foco como seu principal produto, e a empresa passa a ter um maior investimento estratégico em serviços, sobretudo para atender operações em nuvem (RADITS, 2019, p. 32-33). Para consolidar sua nova estratégia, a empresa intensificou sua participação junto a comunidade do *software* de código aberto, onde se destacam:

- A abertura, em 2014, do código-fonte do *framework* de desenvolvimento de *software* .NET Framework, a fim de fortalecer seu ecossistema, interoperabilidade e incentivar adoção para outras plataformas fora do ecossistema Windows — proprietário da Microsoft (RADITS, 2019, p. 33). Destaca-se também a criação da .NET Foundation, organização sem fins lucrativos cujo propósito é promover o ecossistema de código aberto para o *framework*, que passou a ser desenvolvido sob a denominação .NET Core (LARDINOIS, 2014, n.p);
- A participação, em 2015, na fundação da Node.js Foundation, em conjunto com organizações da indústria das quais se destacam a The Linux Foundation, IBM, e Intel. A fundação surgiu com o objetivo de fortalecer o ecossistema do Node.js, *runtime* de código aberto para a execução de aplicações em JavaScript para a *web* (KERNER, 2015, n.p);
- A aquisição, por US$ 7,5 bilhões, em 2018, do GitHub, a maior plataforma de infraestrutura para repositórios de projetos de código aberto até então (DAHLANDER; GANN; WALLIN, 2021, p. 5). No ano anterior, a Microsoft já havia firmado parceria com a empresa para disponibilizar seus projetos de *software*

> de código aberto pela plataforma, como o editor de texto para desenvolvimento integrado Visual Studio Code e o *framework* .NET Core (RADITS, 2019, p. 35).

A aproximação da Microsoft, todavia, não foi um caso isolado na indústria. Durante a virada do século XX para o XXI foi possível notar um grande e repentino interesse de grandes corporações na área de tecnologia da informação e comunicação no *software* de código aberto (LERNER; TIROLE, 2002, p. 197). É possível notar quatro principais fatores por trás desse fenômeno: i) a rápida difusão do *software* de código aberto, com projetos como o servidor *web* Apache, os sistemas operacionais GNU/Linux, o agente de transporte de e-mail Sendmail e a linguagem de *scripting* para *Common Gateway Interface* (CGI) Perl dominando seus respectivos segmentos no mercado; ii) a significativa quantidade de capital investido em projetos de código aberto, com grandes corporações como a HP, IBM e Sun Microsystems lançando projetos para desenvolver e utilizar *software* de código aberto, para além do surgimento de *startups* para comercialização e suporte do ecossistema GNU/Linux como a Red Hat e a VA Linux — ambas as quais realizaram suas ofertas públicas iniciais em bolsas de valores; iii) o surgimento da Open Source Initiative, no final da década de 1990, advogando pelo uso de licenças de código aberto mais permissivas, bem como pelos benefícios práticos do modelo de desenvolvimento aberto e; iv) a difusão da natureza colaborativa do *software* de código aberto, sobretudo impulsionada através da expansão da Internet desde meados da década de 1990, contribuindo para o maior interesse e adoção do modelo (LERNER; TIROLE, 2002, p. 197-200; MEDAPPA; SRIVASTAVA, 2020, p. 1-4).

Lerner e Tirole propõem duas possíveis estratégias para o desenvolvimento colaborativo proveniente do *software* de código aberto ser interfaceado com o desenvolvimento fechado advindo do *software* proprietário: i) a empresa emula alguns dos aspectos do modelo aberto dentro de seu modelo fechado ou ainda; ii) combina os aspectos e processos de ambos os modelos visando tirar proveito de ambos (LERNER; TIROLE, 2002, p. 223). Os autores propõem que empresas que escolhem adotar a primeira estratégia podem replicar certos aspectos inerentes à cultura do modelo colaborativo internamente entre seus funcionários, por exemplo, através de um sistema de incentivos que visa reconhecer e tornar seus funcionários-chaves visíveis a fim de atrair potenciais talentos para si (LERNER; TIROLE, 2002, p. 223-224), ou ainda empregar formas de integrar clientes em seu processo de inovação, por exemplo, através de *crowdsourcing*, customização em massa, integração com a sua comunidade de clientes, ou ainda através de agentes que atuam como intermediários de inovação aberta (ENKEL; GASSMANN; CHESBROUGH, 2009, p. 312). Já para as empresas que optam pela segunda estratégia, há uma ampla gama de caminhos pelos quais

essas empresas podem interagir com comunidades de *software* de código aberto; elas podem optar por uma abordagem reativa, por exemplo, alocando seu próprio pessoal em projetos de código aberto a fim deles incorporarem conhecimento e habilidades externas; ou ainda optar por uma abordagem proativa, por exemplo, licenciando abertamente seu código proprietário (portanto, despejando ou exportando conhecimento) e criando uma estrutura de governança para o processo decorrente (CIESIELSKA; WESTENHOLZ, 2016, p. 345-346).

De acordo com Grand *et al*. (2004, p. 595), no desenvolvimento de *software* de código aberto, as empresas de *software* e a comunidade coexistem através de uma relação simbiótica. Embora cada uma dessas partes possa a vir a ter uma motivação própria diferente para contribuir com seus esforços, todas elas podem se beneficiar do processo de cooperação. As empresas são beneficiadas quando usuários inovam e compartilham suas inovações com elas, pois receberão informações sobre demandas dos usuários e ideias técnicas; por outro lado, os usuários esperam que este ato de cooperação trará a eles melhores produtos, bem como facilitar seu processo de aprendizagem sobre novas tecnologias juntamente com as empresas. Entretanto, para os autores, da perspectiva da empresa, essa cooperação é altamente dinâmica e apresenta diferentes níveis de engajamento para com a comunidade. Os autores propõem que a cooperação da empresa com a comunidade pode ser melhor descrita através de um modelo de análise composto por quatro níveis cumulativos de engajamento; quanto mais alto o nível, maior é a alocação de recursos por parte da empresa no processo e maior é a capabilidade do processo de produção de inovação em beneficiá-la (GRAND *et al*., 2004, p. 592-593). O Quadro 4 relaciona cada nível do modelo com seu respectivo contexto de engajamento corporativo para com o *software* de código aberto.

Quadro 4 – Níveis de engajamento corporativo para com o *software* de código aberto
(continua)

Nível	Perspectiva	Descrição
1	A empresa como usuária de *software* de código aberto.	Neste nível a empresa atua mais como usuária do que desenvolvedora de *software* de código aberto. Ela explora o fato do *software* estar disponível publicamente e o inclui a sua infraestrutura de TI, bem como ocasionalmente fornece *feedbacks* e *patches* de código para a comunidade. Assim como os demais usuários, sua única obrigação legal e formal é estar ciente dos termos de licenciamento do *software*. A motivação da empresa para a inclusão do *software* entre sua infraestrutura de TI está nos potenciais benefícios que nele ela observa, como sofisticação tecnológica, escalabilidade, confiabilidade, efetividade de custo e ausência de aprisionamento tecnológico. O processo pelo qual a empresa passa a ser usuária do *software* de código aberto, entretanto, não é isento de custos; é necessário conhecimento técnico por parte da empresa para instalá-lo e integrá-lo em sua infraestrutura de TI; essas competências precisam ser desenvolvidas internamente ou em colaboração com empresas de *software* especializadas (GRAND *et al*., 2004, p. 595-596).

(conclusão)

Nível	Perspectiva	Descrição
2	O *software* de código aberto como um ativo complementar.	Neste nível, além de realizar os passos anteriores, a empresa passa a perceber o *software* de código aberto como um ativo complementar; ela utiliza o *software* para complementar seus produtos desenvolvidos e produzidos internamente. Consequentemente, a empresa passa a investir mais no *software* de código aberto para adaptá-lo às suas soluções, e isso exige mais engajamento, desenvolvimento e capabilidade inovativa em comparação ao primeiro nível (GRAND *et al*., 2004, p. 596-597).
3	O *software* de código aberto como uma escolha de *design*.	Neste nível, além de realizar os passos anteriores, os altos investimentos realizados pela empresa para incluir o *software* de código aberto em suas soluções de negócio dão a ela a segurança necessária para torná-lo uma escolha de *design*, isto é, o *software* de código aberto passa a ser uma das formas através da qual ela desenvolve novos produtos. A filosofia do movimento colaborativo passa a exercer influência nos processos e na mentalidade da empresa e o *software* em seus negócios e interações com clientes. Neste nível, a empresa contribui ativamente com a comunidade, e isto requer altos investimentos em pesquisa, desenvolvimento, comunicação e coordenação. A empresa passa a disponibilizar o código-fonte desenvolvido internamente com a comunidade, uma vez que ela passa a entender que o seu sucesso econômico não está atrelado ao secretismo do código produzido, mas sim ao conhecimento e expertise necessários para desenvolvê-lo e incrementá-lo, capacidades essas que não são facilmente reproduzidas por concorrentes (GRAND *et al*., 2004, p. 597–598).
4	O *software* de código aberto como o componente central de um modelo de negócio.	Neste último nível, o *software* de código aberto torna-se o componente central no modelo de negócio geral da empresa, pois ela consegue identificar e explorar caminhos para geração de receita e simultaneamente quebrar com a ideia tradicional de modelos baseados na proteção à propriedade intelectual fechada. Há significativa colaboração próxima e contínua com a comunidade através de múltiplos projetos de *software* de código aberto; pelo mesmo motivo, a empresa necessita estar atenta às normas sociais da comunidade. Empresas neste nível também experienciam custos substancialmente menores em desenvolver novos produtos, tendo em vista que ela tem acesso e sabe explorar o potencial de comunidades amplas de desenvolvedores (GRAND *et al*., 2004, p. 598-600).

Fonte: Adaptado de Grand *et al*. (2004, p. 595-600).

Junto à ascensão do *software* de código aberto na indústria, observou-se também o surgimento de inúmeros modelos de negócio com os quais empresas obtêm receita. Perr, Appleyard e Sullivan (2010, p. 434), identificaram empiricamente três importantes fatores de sucesso para modelos de negócios baseados em *software* de código aberto: i) a seleção de um modelo de licenciamento de código, a qual depende de uma estratégia de posse de propriedade intelectual; ii) o gerenciamento da comunidade de desenvolvimento colaborativo através de mecanismos de governança; iii) a habilidade dos gestores em elaborar um modelo de negócio apropriado para os segmentos de mercados e categorias de produtos desejadas.

> As práticas de inovação dos desenvolvedores em companhias de *software* livre e de código aberto seguem uma estratégia que mistura a convencional perspectiva

> capitalista de mercados e consumidores e o processo de inovação não convencional do *software* livre e de código aberto. O padrão de inovação nas empresas de *software* livre e de código aberto é, portanto, um híbrido que busca lucros mas que está profundamente ligado às suas interações com a comunidade de *software* livre e de código aberto (LIN, 2006, p. 89, tradução nossa[8]).

Chesbrough e Appleyard (2007, p. 66) fornecem uma categorização concisa dos modelos de negócio para *software* de código aberto presente na indústria, descrita através do Quadro 5.

Quadro 5 – Principais categorias de modelos de negócios para *software* de código aberto (continua)

Categoria	Modelo	Descrição
Entrega	Consultoria	No modelo baseado em consultoria, a empresa gera receita através da oferta de serviços profissionais para o *software* de código aberto como consultoria, certificações, capacitação profissional, customização e integração para clientes corporativos (PERR; APPLEYARD; SULLIVAN, 2010, p. 443).
	Suporte	No modelo baseado em suporte, a empresa gera receita através de contratos de suporte corporativo para clientes, como documentação *online*, atendimento via e-mail, telefone ou *help desk*. Embora a empresa conte com programadores internos, o desenvolvimento do *software* é confiado a comunidades externas (PERR; APPLEYARD; SULLIVAN, 2010, p. 443-444).
	Assinatura	No modelo baseado em assinatura, a empresa gera receita através de suporte por assinatura para o *software* de código aberto durante um determinado período de tempo. O modelo tira proveito da relativa dificuldade em manualmente testar, integrar e entregar correções e novas funcionalidades para o *software* em ambientes de produção. Essas empresas, portanto, buscam automatizar e assegurar de forma eficiente esse processo nos sistemas de seus clientes (PERR; APPLEYARD; SULLIVAN, 2010, p. 444-445).

[8] No original: “the innovation practices of developers at FLOSS companies follow the strategy which is a mixture of the conventional capitalist perspective on markets and customers, and the unconventional FLOSS innovation procedure. The innovation pattern in FLOSS firms therefore entails a hybrid one that pursues profits but is deeply linked to their interactions with the FLOSS community”.

(conclusão)

Categoria	Modelo	Descrição
Hibridização	Licenciamento duplo	No modelo baseado em licenciamento duplo, a empresa licencia seu *software* simultaneamente através de uma licença de código aberto — tipicamente denominada "versão comunitária" — e uma licença proprietária, de código fechado. Neste modelo, tipicamente a empresa busca gerar receita através da oferta da versão proprietária do *software*, que oferece recursos adicionais, enquanto mantém o *software* aberto através de uma versão de licença *copyleft*, como a GPL, impedido que terceiros repliquem o mesmo modelo de negócio (PERR; APPLEYARD; SULLIVAN, 2010, p. 445-446).
	Extensões proprietárias	No modelo baseado em extensões proprietárias, a empresa busca licenciar seu *software* através de uma licença de código aberto a fim de atrair uma grande base de usuários e desenvolvedores para usá-lo e mantê-lo, enquanto desenvolve internamente extensões ou complementos proprietários para o *software* e os comercializa. Essa tática permite, por exemplo, que ela ofereça — através de extensões — funcionalidades distintas que não são de interesse da comunidade (PERR; APPLEYARD; SULLIVAN, 2010, p. 446-447).
Complementação	Dispositivos	No modelo baseado em dispositivos, a empresa busca embutir *software* de código aberto em dispositivos de *hardware* que ela comercializa visando obter maiores margens de receita e extensibilidade (PERR; APPLEYARD; SULLIVAN, 2010, p. 447).
Self-service	*Community source*	No modelo *community source*, um consórcio de empresas, usuários e/ou instituições de outras naturezas (como universidades e órgãos públicos) contribuem conjuntamente para o desenvolvimento de um *software* de código aberto para si com a finalidade de reduzir custos de licenciamento, tendo em mente que os custos de engajar-se no desenvolvimento do *software* aberto é menor que adquirir alternativas proprietárias (PERR; APPLEYARD; SULLIVAN, 2010, p. 447-448).

Fonte: Adaptado de Chesbrough e Appleyard (2007, p. 66) e Perr, Appleyard e Sullivan (2010, p. 443-448).

3 **ESTUDOS DE CASO**

O capítulo atual apresenta o processo de condução dos estudos de caso realizados. O conteúdo do capítulo é apresentado disposto em três subseções. A subseção 3.1 apresenta a metodologia utilizada para a realização dos estudos de caso. A subseção 3.2 apresenta a descrição do caso de uma cooperativa que produz e comercializa *software* livre; a subseção 3.3 apresenta a descrição do caso de uma empresa *spin-off* acadêmica que pretende comercializar sua pesquisa através de um *software* de código aberto.

3.1 **Metodologia**

A primeira etapa para a condução do estudo de caso envolve a definição de um perfil do entrevistado. Uma vez que é de interesse do trabalho abordar o *software* de código aberto a partir de uma perspectiva mais próxima a gerencial, é definida como fundamental a participação de um indivíduo atualmente empregado em um cargo gerencial ou de liderança administrativa-técnica, fazendo parte do processo de tomada de decisão na empresa. O objetivo dessa decisão é ter acesso a informações provenientes de uma visão holística do ciclo de vida do *software* na empresa, bem como perspectivas além do ambiente interno da empresa em questão.

A segunda etapa consiste na elaboração de um roteiro de entrevista concebido com base na fundamentação teórica levantada pelo trabalho. Busca-se elaborar questões abertas, isto é, perguntas que ofereçam margem ao entrevistado complementar sua resposta da maneira que desejar, permitindo assim ao pesquisador ter acesso a uma maior gama de informações.

A etapa subsequente envolve a seleção do participante para a realização da entrevista, a partir dos seguintes critérios: i) a empresa a qual o participante está associado deve possuir ao menos um projeto de código aberto; iii) o projeto de código aberto em questão deve estar atualmente em desenvolvimento e; iii) o projeto de código aberto em questão deve ser oferecido como uma solução comercial. O pesquisador entra em contato com as empresas por e-mail, prestando informações quanto a sua identificação, instituição a qual está filiado e o propósito da pesquisa. A partir da manifestação de interesse das empresas ALFA e BETA e seus respectivos representantes em participar da pesquisa, é solicitado a cada representante a

sua ciência a anuência no termo de consentimento livre e esclarecido elaborado previamente pelo pesquisador com base na Resolução Nº 466, de 12 de dezembro de 2012, do Conselho Nacional de Saúde.

O participante da empresa ALFA entrevistado é o atual CTO (*chief technology officer*) e um dos sócios fundadores da empresa; ele possui mais de 20 anos de experiência profissional na área de tecnologia da informação, atuando em cargos variados como desenvolvimento de sistemas, consultoria e eventos. O participante da empresa BETA é o responsável pelo desenvolvimento de aplicações móveis de um laboratório de desenvolvimento de uma universidade, liderando uma equipe responsável pelo desenvolvimento de aplicações para *smartphones*, *tablets*, TVs, dentre outros dispositivos. Os nomes de ambas as organizações, bem como dos produtos desenvolvidos por elas, são anonimizados a fim de resguardar a privacidade dos entrevistados.

A etapa de coleta de dados é realizada a partir da execução de uma entrevista por meio de videoconferência com o participante. Com consentimento prévio do participante, a entrevista é gravada digitalmente em formato de áudio a fim de estar disponível para revisão e avaliação pelo pesquisador após a sua realização. Adicionalmente, são coletados dados presentes nos *websites* das empresas, bem como nos repositórios dos códigos-fontes dos *softwares* de código aberto estudados.

Na etapa de análise dos dados, adapta-se como estratégia analítica a construção da explanação, conforme proposta por Yin (2001, p. 140). A escolha da estratégia é realizada tendo em mente o propósito do trabalho — fornecer uma explicação para um fenômeno cujas causas não são compreendidas em sua totalidade. Portanto, o objetivo da construção da explanação para a análise do caso é construir uma explanação sobre ele através de sucessivas iterações com base em sua observação e referencial teórico Yin (2001, p. 141).

3.2 **A cooperativa de software livre ALFA**

A ALFA é uma cooperativa digital de especialistas em desenvolvimento de *software* livre brasileira fundada em fevereiro de 2018. Surgiu no final de um dos eventos promovidos por uma comunidade de tecnologia, a partir da iniciativa de um grupo de desenvolvedores em executarem um projeto *freelancer* de forma cooperativa, com administração colaborativa e democrática, onde todos os sócios possuem o mesmo direito de voz e voto. Na atualidade, a

ALFA conta com 22 sócios, atuando na área de consultoria em tecnologia da informação com foco no setor público.

Embora inicialmente a cooperativa procurou se empenhar em oferecer uma gama maior de serviços ao mercado, como o desenvolvimento de sistemas, aplicações e *websites*, esta tarefa requer demasiado esforço, impossibilita a implantação de estratégias estáveis, e impede que as pessoas tenham uma visão clara da atuação da cooperativa, no geral impedindo seu crescimento, motivo pelo qual recentemente ela iniciou um processo para focar o seu negócio em uma quantidade menor se soluções de negócio.

Uma das principais soluções de negócio oferecidas atualmente pela cooperativa ao mercado é o ASSINADOR, uma aplicação para a plataforma de nuvem privada de código aberto Nextcloud com a finalidade de gerenciar e assinar documentos digitais utilizando criptografia de chave pública. O ASSINADOR é um *software* livre e de código aberto, distribuído sob a licença GNU Affero General Public License (AGPL) versão 3. É uma aplicação *web*, com seu *back end* tendo sido construído utilizando a linguagem de programação PHP, enquanto que sua interface gráfica de usuário tendo sido construída com o *framework* de código aberto Vue.js e a linguagem de programação JavaScript.

O ASSINADOR começou a ser desenvolvido a partir do início de 2020, quando o isolamento social ocasionado pela Pandemia de COVID-19 desencadeou a necessidade de pessoas e organizações migrarem de ambientes físicos para digitais, consequentemente migrando do uso de documentos em papéis para documentos digitais. O ASSINADOR permite ao usuário construir uma infraestrutura de chaves públicas (*public key infrastructure* - PKI) em sua própria instância privada do Nextcloud. Utilizando um certificado digital pessoal — e-CPF ou e-CNPJ — ou o certificado gerado pela própria aplicação, seu usuário consegue assinar documentos digitalmente. A aplicação possibilita ainda a criação de uma autoridade de certificação (*certification authority* - CA) nos padrões estabelecidos pela Infraestrutura de Chaves Públicas Brasileira (ICP-Brasil), atendendo aos critérios nacionais de segurança e assegurando sua validade jurídica.

Concebido desde o princípio para ser uma solução livre, o ASSINADOR é desenvolvido de forma incremental sob a forma de um projeto livre e de código aberto, com o código-fonte da aplicação estando disponível através de um repositório do sistema de controle de versões distribuído Git hospedado na plataforma *web* GitHub. Através do seu repositório, o público tem acesso ao código-fonte da aplicação em sua totalidade. Usuários cadastrados na plataforma podem realizar um *fork* (bifurcação) do repositório, realizar modificações no código-fonte e então submetê-las ao repositório original através de uma solicitação de *pull*

request, bem como podem registrar *issues* — requisições para implementação de eventuais novas funcionalidades ou correções de problemas existentes. Essa característica permitiu que usuários de diferentes localidades contribuíssem de forma voluntária para o projeto; dentre essas contribuições, destaca-se a quantidade de traduções da aplicação, com o ASSINADOR contando atualmente com 96 traduções que englobam não só idiomas mas também variantes regionais deles, possibilitando que a aplicação seja adotada globalmente.

Para além de desenvolver o projeto através do modelo livre e de código aberto, a cooperativa desejou introduzi-lo como um produto no mercado. Todavia, esse processo inicialmente precisou contornar desafios, dentre os quais compreender qual proposta de valor a solução iria oferecer. A inexperiência em *marketing* da recém-formada equipe de sócios tornou introduzir a aplicação como um produto concorrente a plataformas e serviços de assinatura proprietários já bem estabelecidos no mercado numa tarefa desafiadora. Entretanto, a abordagem selecionada foi mostrar ao mercado que o ASSINADOR, para além de uma aplicação para realizar a gestão e assinaturas de documentos digitais, trazia consigo valores inerentes ao modelo do *software* livre e de código aberto, focado ainda em garantir ao usuário: i) soberania tecnológica, onde o usuário é o proprietário e possui total controle sobre a aplicação para utilizá-la, alterá-la, provisioná-la e integrá-la a demais sistemas sem estar limitado a imposições contratuais que restrinjam o seu uso através de um restrito controle da propriedade intelectual do fornecedor; ii) transparência, a partir da auditabilidade do código-fonte da aplicação, mesmo quando a interação com o usuário se dá através de uma rede — conforme garantido pela licença GNU AGPL versão 3. O ASSINADOR então se diferencia das soluções concorrentes proprietárias na medida em que o modelo de código fechado adotado por elas não permite a oferta de tais valores ao usuário. Dessa forma, a oferta do ASSINADOR não se resume a oferecer ao usuário uma aplicação para realizar a gestão e assinatura de documentos digitais, mas também envolvê-lo em sua comunidade.

A implantação e manutenção do ASSINADOR e do Nextcloud numa infraestrutura de tecnologia da informação empresarial requer, todavia, prévio conhecimento técnico. Enquanto certas organizações possuem recursos e capital humano qualificado para realizar essa tarefa, outras não possuem; ou ainda possuem, embora desconhecendo as tecnologias e a arquitetura utilizada pela aplicação. Valendo-se dessa característica, o modelo de negócio utilizado para o ASSINADOR baseia-se em consultoria; através do um contrato de horas de serviço, a cooperativa oferece a implantação, atualização e customização da aplicação conforme as necessidades manifestadas pelo cliente. Acrescenta-se também ao valor negociado em contrato os custos operacionais para a entrega da solução.

Devido às suas características, a cooperativa tem atuado na oferta do ASSINADOR junto ao setor público; o controle dos usuários sobre a aplicação possibilita uma menor dependência de órgãos públicos em infraestruturas de tecnologia da informação externas à jurisdição brasileira; serviços e plataformas norte-americanos, por exemplo, estão sujeitos ao *Clarifying Lawful Overseas Use of Data Act* (CLOUD Act) — lei sancionada em março de 2018 para permitir que as autoridades federais americanas obriguem as empresas de tecnologia sediadas nos Estados Unidos, por meio de mandado ou intimação judicial, a fornecerem os dados solicitados armazenados em servidores, independentemente deles estarem localizados em solo americano ou estrangeiro (RUTHERFORD, 2019, p. 1177-1179). Embora o argumento para defender a nova legislação seja baseado no fortalecimento das instituições legais do país, simultaneamente ela levanta questionamentos acerca de seu possível uso como ferramenta para violar a privacidade e as liberdades civis, em especial as de indivíduos e entidades estrangeiras (RUTHERFORD, 2019, p. 1204). O ASSINADOR, portanto, é ofertado como uma solução para organizações brasileiras estarem unicamente em conformidade com a legislação de seu país e possibilitar que elas armazenem seus dados e de terceiros em território nacional, enquanto que no âmbito individual, permite ao usuário ter total controle sobre o armazenamento e a privacidade de seus dados.

A ALFA também vem se empenhando em desenvolver parcerias com demais organizações como forma de complementar sua atuação no mercado. Em 2020, a cooperativa trabalha de forma conjunta com a cooperativa de saúde GAMA para desenvolver uma plataforma de telemedicina a fim de permitir que profissionais de saúde cooperados e seus clientes realizassem consultas *online* — uma demanda decorrente do isolamento social provocado pela Pandemia de COVID-19. Além da infraestrutura tecnológica, a ALFA forneceu também suporte técnico para manutenção da plataforma e capacitação dos profissionais para utilizá-la. Através da parceria, a GAMA conseguiu aumentar sua receita operacional, reduzir custos com seu consultório físico e reduzir pela metade a quantidade de cancelamento de consultas.

Atualmente a cooperativa se empenha também em disseminar informações e atrair contribuidores para o ASSINADOR através da participação em diversos eventos de tecnologia. A abertura do projeto desempenha um papel crítico para o sucesso desta tarefa, uma vez que possibilita a divulgação da solução sem os interesses financeiros de uma solução de código fechado. Embora a divulgação do trabalho seja essencial para a cooperativa, uma vez que confere a ela maior visibilidade no ambiente externo, ela também está ciente que a

participação nos eventos exige a formação prévia de uma estratégia de divulgação para conquistar os contribuidores e diminuir os custos logísticos envolvidos no processo.

3.3 **A spin-off acadêmica BETA**

A BETA é uma empresa *spin-off* acadêmica originada dentro de uma universidade pública brasileira. Seus membros realizam pesquisas, desenvolvem e executam projetos na área de tecnologia da informação para atender às necessidades da comunidade interna de sua universidade, bem como a comunidade externa a ela — em parceria com Estados, Municípios, demais instituições de ensino e pesquisa nacionais e estrangeiras.

Um dos projetos em desenvolvimento pelos membros da comunidade acadêmica é o AUTO AI, que visa implementar um sistema de código aberto para realizar o monitoramento de frotas veiculares. Através da coleta e do armazenamento em massa de dados provenientes dos sensores eletrônicos de bordo de veículos, utilizando o protocolo OBD-II (*on-board diagnostics*), o projeto visa empregar a inteligência artificial para identificar padrões de uso e desgaste dos veículos, possibilitando que eventuais problemas e avarias mecânicas sejam evitadas através de manutenção preventiva, consequentemente gerando redução de custos com manutenção automotiva e ampliando a disponibilidade dos veículos. Esses dados mecanismos serão ainda complementados por dados de geolocalização, e através de um dispositivo móvel presente nos veículos, o sistema permitiria ainda seus condutores terem acesso às análises e comunicarem-se diretamente com a central de comando, bem como gravar digitalmente em áudio e vídeo o ambiente interno e externo ao veículo para execução de análises.

O objetivo do projeto faz com que a solução apresentada por ele seja de interesse de outros órgãos públicos em âmbito nacional e internacional, dado que a solução pode ser generalizada para múltiplos casos de uso que incorporem o controle de uso veicular ou ainda o gerenciamento de rotas. Optou-se por desenvolver o AUTO AI através de um modelo de código aberto — licenciado sob a GNU GPL versão 3 — como forma de envolver a comunidade acadêmica interna, bem como atores no ambiente externo à universidade, formando desta forma uma comunidade de usuários e desenvolvedores capaz de potencializar seu processo de inovação.

Os potenciais impactos sociais e econômicos notados através do AUTO AI motivou o surgimento de uma iniciativa para levá-lo ao mercado, sob a forma de uma *spin-off* acadêmica

— uma *startup* fundada por estudantes e pesquisadores universitários a fim de implementar no mercado produtos idealizados a partir de seus trabalhos de pesquisa acadêmica (SHANE, 2004, p. 4). Através da *spin-off* BETA, o AUTO AI será ofertado ao mercado através de um modelo de negócio baseado em consultoria, capacitação e suporte. Organizações que incorporarem o sistema poderão ainda contratar o desenvolvimento dedicado e priorizado de novas funcionalidades para a base de código do sistema. A empresa será responsável por centralizar o gerenciamento do ecossistema aberto do sistema, incorporando funcionalidades e correções realizadas por desenvolvedores da comunidade e modificações com base em *feedbacks* de usuários.

Uma das fases do projeto consiste na definição de qual licença de código aberto utilizar para regir o desenvolvimento do AUTO AI. Enquanto que a empresa busca empregar o código aberto como uma forma de abrir o processo de inovação relacionado ao projeto, ela também está ciente que, ao fazê-lo, também está sujeita ao risco de não ser inteiramente capaz de se apropriar do valor da inovação presente no projeto, oferecendo margem para que eventuais empresas concorrentes capitalizem o projeto e consequentemente a tornando menos competitiva no mercado. Portanto, a escolha de uma licença mais restritiva ou *copyleft* para o AUTO AI — como a GNU GPL — visa inibir que demais agentes no mercado criem uma solução fechada a partir dele.

Atualmente, o BETA atua na difusão do projeto e na busca de parcerias com instituições de ensino superior para finalizar a implementação do projeto, formando assim uma rede de colaboração para o seu processo de desenvolvimento. Através dessa ação, o projeto conseguiu identificar uma iniciativa similar ao AUTO AI sendo executada no estado do Mato Grosso do Sul, utilizando redes Wi-Fi de longo alcance para comunicação móvel; surgiu assim a proposta de adicionar essa característica ao próprio AUTO AI — que emprega o uso de redes de telefonia móvel — a fim servir de tornar a comunicação móvel do sistema mais resiliente a falhas.

4 **DISCUSSÃO DOS RESULTADOS DAS ENTREVISTAS**

A ALFA é um caso particular de inovação aberta na indústria de *software* brasileira. Ao formarem uma cooperativa, seus sócios aderiram a uma forma de organização da atividade econômica em grande parte similar aos princípios do modelo de desenvolvimento colaborativo presente em comunidades de *software* livre e de código aberto. Na definição da Aliança Cooperativa Internacional (*International Cooperative Alliance* - ICA):

> Uma cooperativa é uma associação autônoma de pessoas unidas voluntariamente para atenderem suas necessidades e anseios econômicos, sociais e culturais comuns através de uma empresa de propriedade conjunta e democraticamente controlada (ICA, 2024, n.p, tradução nossa[9]).

Assim como o desenvolvimento cooperativo de *software*, o trabalho é executado e organizado de forma horizontalizada. Enquanto o presente trabalho focou-se em abordar a adoção estratégica do código aberto na indústria sob um viés gradual, onde as empresas são estruturadas de forma tipicamente hierárquica, com visões, valores e missões distintos aos que as formas cooperativas de trabalho propõem, a ALFA teve o trabalho cooperativo e seus princípios como um de seus pilares em sua fundação; todavia, essa característica do caso não impediu a identificação das práticas de inovação aberta na empresa. A *spin-off* BETA, por sua vez, surge da percepção da necessidade em levar o fruto do trabalho de pesquisa acadêmica ao mercado, produzindo impactos na sociedade. Ela vale-se dos princípios baseados na publicidade e difusão do conhecimento — sem ausência de dono e barreiras de acesso; bem como dois níveis de troca de conhecimento, sendo um deles composto pela comunidade e o outro por um grupo de especialistas.

Tendo surgido a menos de uma década atrás no contexto dos rápidos avanços tecnológicos que dominam o mercado de *software*, a ALFA foi fundada e estruturada a partir dos princípios da cooperação; de forma similar, a *spin-off* BETA surge no contexto do ambiente e da pesquisa acadêmica, onde há intrinsecamente o incentivo à cooperação como forma de desempenhar o trabalho e alcançar resultados. A cooperação estruturam não apenas a forma com a qual essas empresas se organizam, mas também suas práticas e seus processos — incluindo o de inovação. A prática da inovação aberta na ALFA e na BETA — através do desenvolvimento de *software* de código aberto —, portanto, aparenta ser decorrente da própria

[9] No original: “A cooperative is an autonomous association of persons united voluntarily to meet their common economic, social and cultural needs and aspirations through a jointly-owned and democratically-controlled enterprise”.

natureza cooperativa do ambiente de produção de conhecimento tecnológico onde essas empresas estão inseridas.

Como prática de inovação aberta *inbound* (ver Subseção 2.4.2), a cooperativa ALFA realiza a incorporação de propriedade intelectual externa, como evidenciado pela inclusão da plataforma Nextcloud em seu portfólio de soluções de negócio. Além disso, a empresa incorpora conhecimento externo ao participar ativamente do desenvolvimento do ASSINADOR, promovendo contribuições de agentes externos para o código-fonte da aplicação mantida por ela. Ao incorporar o conhecimento difundido em comunidades de *software* livre e de código aberto nas quais proativamente a cooperativa participa, ela tem acesso a uma ampla gama de ideias ou *inputs* para complementar o seu processo de inovação interno, possibilitando a oferta de produtos e serviços ao mercado mais diversificados e inovadores; especificamente no caso estudado, a cooperativa inovou ao oferecer uma solução de negócio com uma proposta de valor única no mercado unindo os valores do cooperativismo e do modelo cooperativo de desenvolvimento de *software*. Na *spin-off* BETA, a inovação aberta *inbound* se manifesta no amplo emprego da colaboração externa com outras universidades, bem como nas práticas colaborativas do modelo de código aberto como forma de complementar o processo de desenvolvimento do AUTO AI, enquanto almeja buscar incluir o próprio cliente nesse processo.

De forma complementar, como prática de inovação aberta *outbound* (ver Subseção 2.4.2), ao exteriorizar o seu conhecimento interno para as comunidades, através da revelação do código-fonte desenvolvido internamente, tanto a cooperativa ALFA como a *spin-off* BETA buscam fomentar o crescimento e a capacidade inovativa dessas comunidades, as tornando sustentáveis através do trabalho colaborativo entre os agentes que nela estão inclusos e buscando com isso realimentar o seu processo de inovação enquanto aprimora o valor das soluções de negócio oferecidas através de seus respectivos modelos de negócio baseados em consultoria e suporte. Ademais, ambas as empresas estudadas também buscam difundir o seu conhecimento interno ao participar ativamente de eventos promovendo a adoção de seus projetos.

Como prática de inovação aberta *coupled* (ver Subseção 2.4.2), destaca-se que em ambos os casos estudados os agentes buscam desenvolver uma forma de colaboração com organizações externas. Enquanto que através do modelo de inovação fechada a ALFA seria responsável por todo o processo de concepção e entrega da solução tecnológica ao mercado — assumindo um eventual risco dela não ser bem recebida pelos usuários —, ao firmar parceria com a cooperativa de saúde a ALFA buscou atuar de forma cooperativa no

desenvolvimento da plataforma de telemedicina demandada, valendo-se dos princípios de intercooperação do cooperativismo; a parceria resultou em uma solução que atendeu as demandas solicitadas e prontamente foi aceita e utilizada. A *spin-off* BETA, por sua vez, deseja modelar sua solução de negócio conforme as necessidades levantadas pela sua comunidade de usuários. A partir do envolvimento de agentes externos no processo de inovação de forma bidirecional, ambos os casos subvertem a noção tradicional de mercado composta por uma relação unidirecional da inovação que surge na empresa e é recebida pelo consumidor. O Quadro 6 relaciona os mecanismos necessários para um modelo de negócios (ver Subseção 2.4.2), conforme proposto por Chesbrough (2003b, p. 64), com aqueles encontrados no modelo utilizado pela cooperativa para o ASSINADOR.

Quadro 6 – Mecanismos dos modelos de negócio utilizados pelas empresas estudadas

Mecanismo	ALFA	BETA
Proposição de valor	Oferecer soberania, independência e transparência tecnológica para seus usuários através da promoção e incentivo ao uso de *software* livre e de código aberto.	Oferecer uma solução tecnológica de ponta para seus usuários, possibilitando menor custo com manutenção de frotas veiculares.
Segmento de mercado	Setores públicos (em especial educacional) e privados (pequenas e médias empresas).	Setor público em geral.
Cadeia de valor	Composta pelo desenvolvimento proativo de *software* livre e de código aberto suportada por divulgações de *marketing*.	Composta pelo desenvolvimento proativo de *software* de código aberto em colaboração com instituições acadêmicas e órgãos públicos.
Geração de receita	Consultoria composta por suporte, customização e capacitação técnica.	
Rede de valor	Formada pela relação de cooperação com demais cooperativas no mercado, comunidade de desenvolvedores e comunidades de *software* livre e de código aberto.	Formada pela relação de cooperação com instituições de ensino superior e promoção a pesquisa, órgãos públicos e comunidades de *software* de código aberto.
Estratégia competitiva	Inovar a partir da oferta de produtos que trazem consigo os valores do cooperativismo e do *software* livre e de código aberto.	Inovar a partir da oferta de produtos que empregam tecnologia de ponta desenvolvida de maneira colaborativa.

Fonte: elaborado pelo autor.

Entretanto, é interessante destacar que ambas as empresas estudadas possuem visões distintas do papel do código aberto em seu ambiente empresarial. Enquanto que na ALFA a finalidade do código aberto está mais associada à ética do *software* livre — através da promoção dos direitos fundamentais de uso, estudo, modificação e redistribuição —, na BETA o código aberto é visto de maneira mais pragmática, como uma ferramenta, método ou

modelo para tornar o *software* melhor e o trabalho de desenvolvimento mais eficiente. Todavia, essas duas maneiras de se perceber o código aberto não são mutuamente exclusivas dado que ambas as empresas concordam com as duas premissas, embora optem por priorizar somente uma delas de acordo com suas finalidades. Essas visões distintas presentes em ambas as empresas estudadas podem ser explicadas através da familiaridade de suas respectivas culturas empresariais com o modelo de código aberto: na ALFA, seus sócios fundadores, além de já contarem com prévia experiência contribuindo para projetos de código aberto, demonstraram confiar no modelo tanto quanto o suficiente para torná-lo central na cultura da empresa, bem como em quaisquer futuros projetos; na BETA, onde o modelo de código aberto não desempenha uma competência fundamental para a empresa, ele é tido como mais uma opção para desenvolver *software* dentre as já existentes e o seu emprego em um novo projeto necessita antes definido pela ponderação entre os seus potenciais benefícios e riscos associados.

5 CONSIDERAÇÕES FINAIS

Através do presente trabalho, buscamos contribuir para os estudos de sistemas de informação procurando analisar o *software* de código aberto dentro do ambiente empresarial das empresas de tecnologia da informação, almejando fornecer uma explicação clara e concisa das razões pelas quais empresas da indústria de *software* optam por conferir ao *software* de código aberto um *status* de recurso estratégico para alcance de seus objetivos estratégicos. Fornecer uma explicação para esse fenômeno exige, entretanto, uma prévia compreensão da estrutura da indústria — em especial os mecanismos de apropriabilidade de valor da inovação — e sua contextualização histórica — abordando os fatos que a moldaram no decorrer dos anos.

A produção de *software* no período inicial da computação digital surge no contexto acadêmico, caracterizado pela cooperação como forma de produção de conhecimento e inovação. A partir da década de 1960, o interesse do mercado em obter capital através do *software* introduz uma nova concepção que retira dele a agregação cooperativa do conhecimento de múltiplos indivíduos e o transforma em um produto de mercado. Esse movimento é decorrente do entendimento que as redes de colaboração não são capazes de se apropriar do valor da inovação, que naquele contexto deveria ser realizada pelo mercado — na figura das empresas privadas — através da proteção legal à propriedade intelectual. Entretanto, o avanço das redes de cooperação pela sociedade, oferecendo a divisão de custos e riscos, bem como maior capacidade de inovação tecnológica, se sobrepôs ao interesse do mercado. Dentro das redes, novas oportunidades são criadas a todo momento, tornando a sobrevivência fora delas uma tarefa cada vez mais árdua; dessa forma, as redes, e não mais as empresas, se tornam os principais agentes da nova economia (CASTELLS, 2002, p. 232). As transformações nas maiores empresas ao longo das duas últimas décadas do século XX não representam um novo e melhor modo de produção, mas a crise de um modelo antigo e poderoso, excessivamente rígido, caracterizado também pela sua verticalidade e controle oligopolista dos mercados (CASTELLS, 2002, p. 224).

> [...] a economia global agora é uma rede de segmentos econômicos interconectados, que, juntos, têm um papel decisivo na economia de cada país — e de muitas pessoas. Depois de constituída tal rede, qualquer nó que se desconecte é simplesmente ignorado, e os recursos (capital, informações, tecnologia, bens, serviços, mão-de-obra qualificada) continuam a fluir no resto da rede. Qualquer indivíduo que se afaste da economia global acarreta custos elevadíssimos: a devastação da economia em curto prazo e o bloqueio do acesso às fontes de desenvolvimento.

> Assim, dentro do sistema de valores do produtivismo/consumismo, não há alternativa individual para países, empresas ou pessoas (CASTELLS, 2002, p. 188-189).

A partir da literatura disponível — sobretudo nas áreas de inovação, computação, informação e administração — concluímos que a adoção estratégia do *software* de código aberto nas empresas de desenvolvimento de *software* é uma das formas com as quais elas buscam manter-se competitivas num mercado altamente dinâmico — caracterizada pelo rápido ritmo de inovação tecnológica — através da cooperação em redes como forma de fomento a atividade de inovação. Ao incorporarem o conhecimento disponível em comunidades de *software* livre e/ou de código aberto, através de processos *inbound* de inovação aberta, essas empresas têm acesso a uma ampla gama de ideias ou *inputs* para complementar seus processos de inovação interno, possibilitando a oferta de produtos e serviços ao mercado mais diversificados e inovadores. Ao exteriorizarem o seu próprio conhecimento internos para as comunidades, através de processos *outbound* de inovação aberta, essas empresas buscam fomentar o crescimento e capacidade inovativa dessas comunidades, buscando com isso obter indiretamente benefícios para seu processo de inovação enquanto podem explorar o valor das inovações produzidas através de um modelo de negócio.

> [...] é interessante essa reflexão para tirarmos essa barreira de que o *software* proprietário é o *software* que vai dar dinheiro e o *software* livre é o *software* que não pode ser monetizado [...] quando você abre o código, você consegue potencializar as pessoas que vão analisar aquele código e contribuir para que ele fique melhor [...] quando eu construo algo de forma coletiva, conseguimos sim fazer uma transformação na sociedade, aí sim que conseguimos trazer um impacto relevante (participante da empresa ALFA entrevistado).

> [...] o *software* aberto [...] é a representação na área de *software* de como nós produzimos conhecimento na sociedade (participante da BETA entrevistado).

Na relação entre o ambiente empresarial e as comunidades de *software* de código aberto, em especial, a cooperação tem espaço para especial atenção, sendo favorecida pelo contexto tecnológico formado pela difusão facilitada do conhecimento. Empresas que desejam abrir o seu processo de inovação através do *software* de código aberto necessitam antes compreender o real valor e significado da cooperação, a fim de não tornar o processo numa relação onde apenas ela se beneficia do trabalho de todos os agentes envolvidos. Isso envolve, antes, uma mudança na visão em como conduzir o processo de inovação; se no modelo de desenvolvimento de código fechado as redes de colaboração baseadas no código aberto são

vistas por determinadas empresas com desconfiança ou até mesmo como uma ameaça à sobrevivência delas no mercado, para as empresas que optaram por abrir seu processo de inovação essas mesmas redes de colaboração são vitais.

> [...] Sempre tentamos chamar a comunidade para assumir a responsabilidade de que se ela está utilizando um *software* livre então é importante que de alguma forma ela contribua para esse *software* [...] a comunidade de *software* livre está aí para isso — você fazer parte dela —, não pra você usá-la, mas para você ser membro e contribuir para o crescimento da comunidade, porque estamos crescendo de forma coletiva (participante da empresa ALFA entrevistado).

A partir daí, o envolvimento da empresa nessas comunidades precisa se dar de forma proativa; deve-se buscar não só atrair contribuidores ou demais agentes para o processo, mas também em como mantê-lo sustentável ao decorrer do tempo (CHESBROUGH; APPLEYARD, 2007, p. 67-68). A inovação aberta pode, então, ser compreendida como uma atividade contínua, requerendo uma gestão estratégica e o amplo envolvimento daqueles com poder de tomada de decisão na empresa.

A importância das comunidades e seu modelo de produção da inovação, bem como suas iniciativas através de projetos de código aberto, tem atraído cada vez mais a atenção de grandes e influentes corporações de tecnologia da informação atuantes no mercado. Em 2016, a The Linux Foundation, organização sem fins lucrativos responsável pela formação de um consórcio composto por 800 entidades unidas para promover, padronizar e proteger o desenvolvimento do *kernel* Linux, removeu os assentos independentes no seu conselho administrativo, eliminando a disposição que permitia a eleição para o conselho de dois membros filiados a grupos independentes. Dessa forma, o conselho administrativo da função passou a ser completamente eleito pelas entidades corporativas de nível “platina” — as que pagam anualmente US$ 500 mil à fundação para serem incluídas em seu conselho administrativo. Esse seleto grupo de tomadores de decisão é composto por grandes corporações como a Google, AT&T, Cisco, Fujitsu, Hitachi, Huawei, IBM, Intel, Microsoft, NEC, Oracle, Qualcomm, Samsung, e VMware (DAHLANDER; GANN; WALLIN, 2021, p. 6). Em 2018, a Microsoft realizou a aquisição, por US$ 7,5 bilhões, do GitHub — maior plataforma de hospedagem e desenvolvimento de *software* de código aberto até então, englobando cerca de 50 milhões de desenvolvedores operando 85 milhões de repositórios de código-fonte aberto e utilizado por mais de 1,5 milhão de empresas — representando um grande polo de inovação aberta na indústria e na *web*. Esse movimento, entretanto, foi recebido por grande parte da comunidade com preocupação, dada a posição contrária histórica

da empresa quanto ao modelo de código aberto (DAHLANDER; GANN; WALLIN, 2021, p. 5-6). Outro grande importante movimento no mercado se deu em julho de 2019, quando a IBM concluiu a aquisição da Red Hat por US$ 34 bilhões, a posicionando como líder no mercado de provedores de nuvem híbrida global (DAHLANDER; GANN; WALLIN, 2021, p. 6).

Em anos recentes, é possível observar o surgimento de casos de conflito entre o modelo de código aberto e a sua corporatização pela indústria, com destaque especial para o mercado de *software* como serviço (*software as a service* — SaaS). Em outubro de 2018, por exemplo, a MongoDB Inc., responsável pelo desenvolvimento do banco de dados não relacional (ou NoSQL) MongoDB, alterou a licença do projeto da AGPL versão 3 para a Server Side Public License (SSPL), uma licença elaborada pela própria empresa. A empresa justificou a medida como uma resposta às práticas de demais empresas no mercado em oferecer serviços concorrentes baseados no MongoDB e não contribuírem para o desenvolvimento colaborativo da plataforma (MONGODB INC, 2018, n.p). A licença SSPL, entretanto, não é considerada, pelas definições da FSF e da OSI, como uma licença de *software* livre e/ou de código aberto, uma vez que limita e discrimina o uso do *software* ao exigir que terceiros que desejem oferecê-lo sob a forma de um produto ou serviço comercial obtenham uma autorização prévia concedida pelo detentor original dos direitos autorais sobre o *software*. Em outro caso similar, ocorrido em janeiro de 2021, a Elastic NV — responsável pelo desenvolvimento do motor de busca analístico de código aberto Elasticseach e pela plataforma de visualização de dados Kibana — acusou a Amazon de oferecer o Amazon Elasticsearch Service — uma solução de negócio para a nuvem Amazon Web Services (AWS) — baseada nos serviços de código aberto desenvolvidos por ela, prejudicando sua própria operação no mercado dada a posição dominante da Amazon no mercado de serviços em nuvem. A resposta dada pela Elastic foi migrar o licenciamento de seus projetos da licença Apache License 2.0 para a SSPL (VAUGHAN-NICHOLS, 2021, n.p; DAHLANDER; GANN; WALLIN, 2021, p. 6). A Amazon posteriormente realizou um *fork* dos projetos da Elastic e começou a desenvolvê-lo por conta própria, sendo ofertado sob o nome Amazon OpenSearch. Estes exemplos sinalizam que a relação entre a inovação aberta e a literatura sobre estratégia e dinâmica competitiva precisa ser melhor elucidada, dado que demonstram que os pioneiros a inovarem não serão, necessariamente, os que irão deter os lucros provenientes do valor da inovação (DAHLANDER; GANN; WALLIN, 2021, p. 6).

Tendo observado esses importantes eventos, Dahlander, Gann e Wallin (2021, p. 5-6) questionam se, com uma corporatização crescente do trabalho comum — representado aqui

pelo *software* de código aberto — o que permanecerá em aberto e como isso irá alterar os padrões de contribuição das pessoas que compartilham conhecimento voluntariamente? Para os autores, a falta de dados e informações empíricas no estudo da inovação aberta ainda não permitem uma elucidação da questão proposta, mas concordam que a corporatização do trabalho comum trará novos desafios para a forma de se pensar a inovação — tanto em aspectos de mercado quanto sociais. Enquanto o envolvimento corporativo no mundo do código aberto pode ser benéfico, ele também é pensado como um agente capaz de prejudicar a sustentabilidade dessas comunidades, na medida em que modifica as relações sociais de todas as partes interessadas.

6 **CONCLUSÃO E TRABALHOS FUTUROS**

A produção do presente trabalho é motivada pela observação da expansão do *software* de código aberto na indústria simultaneamente junto a dificuldade de se compreender as razões pelas quais empresas do setor de tecnologia da informação optam por integrá-lo estrategicamente em seus modelos de negócios. Para compreender como essas empresas se apropriam do valor criado através da inovação em bens públicos em uma economia capitalista fortemente marcada pela concorrência e dinamicidade, o presente trabalho propõe o uso da inovação aberta, proposta por Henry W. Chesbrough em 2003, como forma de condução rápida e responsiva para o processo de inovação no desenvolvimento de *software* na atualidade. O trabalho apresenta uma análise histórica e atual da indústria suportada por dois estudos de casos de padrões de inovação conduzidos em duas empresas brasileiras do setor de tecnologia da informação. Os resultados obtidos revelam que as empresas buscam se apropriar do valor da inovação pública presente no modelo de código aberto através de diferentes métodos como forma de se inserir e diferenciar-se no mercado. Demonstramos que a integração estratégica do *software* de código aberto e o seu modelo de desenvolvimento colaborativo por parte de empresas privadas não se limita a casos isolados ou experimentais na indústria, mas sim a um movimento pressionado por uma nova forma de se inovar a partir de redes de conhecimento, as quais incluem as inúmeras comunidades de *software* de código aberto que hoje assumem papéis cada vez mais proeminentes na indústria. Este estudo conclui que a inovação aberta demonstra ser uma abordagem de pesquisa sólida e adequada para elucidar as razões para a integração estratégica do *software* de código aberto nos modelos de negócio das empresas de tecnologia da informação.

Enquanto que o presente trabalho buscou demonstrar as razões pelas quais empresas do setor de tecnologia da informação adotam o *software* livre e/ou de código aberto em seus modelos de negócio a partir da perspectiva da inovação aberta, ele também é resultado de uma percepção de uma tendência mundial na indústria e disserta sobre a inovação aberta ao nível da indústria, motivo pelo qual não buscou-se através dele explorar especificidades geográficas, assumiu-se ainda uma indústria e mercado com características homogêneas. É evidente que empresas não existem de forma isolada no mercado, e conforme Chesbrough, Vanhaverbeke e West (2006, p. 286-287) propõem, a inovação aberta pode ser compreendida em cinco níveis de análise: o indivíduo, a organização, a rede de valor, a indústria/setor e as instituições nacionais. Os resultados encontrados através do presente trabalho instigam,

portanto, trabalhos futuros a abordar e discutir a inovação aberta através do *software* de código aberto a partir dos diferentes níveis de análise propostos, a fim de enriquecer a abordagem de pesquisa.

REFERÊNCIAS

AGUSTINHO, E. O.; GARCIA, E. N. Inovação, Transferência de Tecnologia e Cooperação. **Direito e Desenvolvimento**, v. 9, n. 1, p. 223–239, 2018. Disponível em: <https://doi.org/10.25246/direitoedesenvolvimento.v9i1.525>. Acesso em: 28 jun. 2024.

AKERA, A. Voluntarism and the Fruits of Collaboration: The IBM user group, Share. **Technology and Culture**, v. 42, n. 4, p. 710–736, out. 2001. Disponível em: <https://doi.org/10.1353/tech.2001.0146>. Acesso em: 28 jun. 2024.

AMDAHL, G. M.; BLAAUW, G. A.; BROOKS, F. P. Architecture of the IBM System/360. **IBM Journal of Research and Development**, v. 8, n. 2, p. 87–101, 1964. Disponível em: <https://doi.org/10.1147/rd.82.0087>. Acesso em: 28 jun. 2024.

ANDRADE, E.; TIGRE, P. B.; SILVA, L. F.; SILVA, D. F.; MOURA, J. A. C. de; OLIVEIRA, R. V. de; SOUZA, A. Propriedade Intelectual em Software: O que podemos apreender da experiência internacional? **Revista Brasileira de Inovação**, v. 6, n. 1, p. 31, 2009. Disponível em: <https://doi.org/10.20396/rbi.v6i1.8648940>. Acesso em: 28 jun. 2024.

BELL, G. STARS: Rise and fall of minicomputers [scanning our past]. **Proceedings of the IEEE**. Institute of Electrical and Electronics Engineers, v. 102, n. 4, p. 629–638, 2014. Disponível em: <https://doi.org/10.1109/JPROC.2014.2306257>. Acesso em: 28 jun. 2024.

BIGLIARDI, B.; GALATI, F. Which Factors Hinder the Adoption of Open Innovation in SMEs? **Technology Analysis and Strategic Management**, v. 28, n. 8, p. 869–885, 2016. Disponível em: <https://doi.org/10.1080/09537325.2016.1180353>. Acesso em: 28 jun. 2024.

BLIND, K.; BÖHM, M.; GRZEGORZEWSKA, P.; KATZ, A.; MUTO, S.; PÄTSCH, S.; SCHUBERT, T. **The Impact of Open Source Software and Hardware on Technological Independence, Competitiveness and Innovation in the EU Economy, Final Study Report**. 1. ed. Luxemburgo: Publications Office European Communities/Union (EUR-OP/OOPEC/OPOCE), 2021. Disponível em: <https://doi.org/10.2759/430161. ISBN: 9789276309802>. Acesso em: 28 jun. 2024.

BONACCORSI, A.; GIANNANGELI, S.; ROSSI, C. Entry Strategies Under Competing Standards: Hybrid business models in the open source software industry. **Management Science**, v. 52, n. 7, p. 1085–1098, 2006. Disponível em: <https://doi.org/10.1287/mnsc.1060.0547>. Acesso em: 28 jun. 2024.

CAMPBELL-KELLY, M. Development and Structure of the International Software Industry – 1950-1990. **Business and Economic History**, v. 24, n. 2, p. 74–110, 1995. ISSN: 0849-6825.

CAMPBELL-KELLY, M. Not All Bad: An historical perspective on software patents. **Michigan Telecommunications and Technology Law Review**, v. 11, n. 2, p. 191–248, 2005. ISSN: 1528-8625. Disponível em: <https://repository.law.umich.edu/mttlr/vol11/iss2/2/>. Acesso em: 28 jun. 2024.

CAMPBELL-KELLY, M.; ASPRAY, W. F.; YOST, J. R.; TINN, H.; CON DÍAZ, G. **Computer: A history of the information machine**. New York: Routledge, 4. ed., 394p,

2023. ISBN: 978-1-032-20343-0. Disponível em: <https://doi.org/10.4324/9781003263272>. Acesso em: 28 jun. 2024.

CARGILL, C. F. Evolution and Revolution in Open Systems. **StandardView**, v. 2, n. 1, p. 3–13, 1994. Disponível em: <https://doi.org/10.1145/224145.224146>. Acesso em: 28 jun. 2024.

CARLOTTO, M. C.; ORTELLADO, P. Activist-driven Innovation: Uma história interpretativa do software livre. **Revista Brasileira de Ciências Sociais**, v. 26, n. 76, p. 77–102, 2011. Disponível em: <https://doi.org/10.1590/s0102-69092011000200005>. Acesso em: 28 jun. 2024.

CASTELLS, M. **Sociedade em Rede**. 6. ed. São Paulo: Paz e Terra, 2002. ISBN: 9788521903291.

CERUZZI, P. E. **A History of Modern Computing**. 2. ed. Londres: MIT Press, 2003. 459 p. ISBN 0262532034.

CHESBROUGH, H. W. **Open Innovation: The new imperative for creating and profiting from technology**. 1. ed. Boston: Harvard Business Review Press, 2003b. 272 p. ISBN: 9781578518371.

CHESBROUGH, H. W. The Logic of Open Innovation: Managing intellectual property. **California Management Review**, v. 45, n. 3, p. 33–58, 2003a. Disponível em: <https://doi.org/10.2307/41166175>. Acesso em: 28 jun. 2024.

CHESBROUGH, H. W.; APPLEYARD, M. M. Open Innovation and Strategy. **California Management Review**, v. 50, n. 1, p. 57–76, 2007. Disponível em: <https://doi.org/10.2307/41166416>. Acesso em: 28 jun. 2024.

CHESBROUGH, H. W.; VANHAVERBEKE, W.; WEST, J. (EDS.). **Open Innovation: Researching a new paradigm**. Londres: Oxford University Press, 2006. ISBN: 9780199290727.

CHOI, N.; PRUETT, J. A. The Characteristics and Motivations of Library Open Source Software Developers: An empirical study. **Library & Information Science Research**, v. 37, n. 2, p. 109–117, 2015. Disponível em: <https://doi.org/10.1016/j.lisr.2015.02.007>. Acesso em: 28 jun. 2024.

CIESIELSKA, M.; WESTENHOLZ, A. Dilemmas Within Commercial Involvement in Open Source Software. **Journal of Organizational Change Management**, v. 29, n. 3, p. 344–360, 2016. Disponível em: <https://doi.org/10.1108/jocm-04-2013-0058>. Acesso em: 28 jun. 2024.

COCKBURN, I. M.; MACGARVIE, M. J. Entry and Patenting in the Software Industry. **Management Science**, v. 57, n. 5, p. 915–933, 2011. Disponível em: <https://doi.org/10.1287/mnsc.1110.1321>. Acesso em: 28 jun. 2024.

COHEN, N. B. Symposium: Software as a Commodity: International licensing of intellectual property. **Brooklyn Journal of International Law**, v. 26, n. 1, p. 3–4, 2000. Disponível em: <https://brooklynworks.brooklaw.edu/faculty/656/>. Acesso em: 28 jun. 2024.

DAHLANDER, L.; GANN, D. M.; WALLIN, M. W. How Open is Innovation? A retrospective and ideas forward. **Research Policy**, v. 50, n. 4, p. 104218, 2021. Disponível em: <https://doi.org/10.1016/j.respol.2021.104218>. Acesso em: 28 jun. 2024.

DAHLANDER, L.; GANN, D. M. How Open is Innovation? **Research Policy**, v. 39, n. 6, p. 699–709, 2010. Disponível em: <https://doi.org/10.1016/j.respol.2010.01.013>. Acesso em: 28 jun. 2024.

DÍAZ, G. C. The Text in the Machine: American copyright law and the many natures of software, 1974–1978. **Technology and Culture**, v. 57, n. 4, p. 753–779, 2016. Disponível em: <https://doi.org/10.1353/tech.2016.0106>. Acesso em: 28 jun. 2024.

DUPARC, E.; MÖLLER, F.; JUSSEN, I.; STACHON, M.; ALGAC, S.; OTTO, B. Archetypes of Open-source Business Models. **Electronic Markets**, v. 32, n. 2, p. 727–745, 2022. Disponível em: <https://doi.org/10.1007/s12525-022-00557-9>. Acesso em: 28 jun. 2024.

DZAMASHVILI FOGELSTRÖM, Nina; GORSCHEK, Tony; SVAHNBERG, Mikael; OLSSON, Peo. The Impact of Agile Principles on Market-driven Software Product Development. **Journal of Software Maintenance and Evolution: Research and Practice**, v. 22, n. 1, p. 53–80, 2010. Disponível em: <https://doi.org/10.1002/smr.453>. Acesso em: 28 jun. 2024.

EDISON, H.; BIN ALI, N.; TORKAR, R. Towards Innovation Measurement in the Software Industry. **The Journal of Systems and Software**, v. 86, n. 5, p. 1390–1407, 2013. Disponível em: <https://doi.org/10.1016/j.jss.2013.01.013>. Acesso em: 28 jun. 2024.

ENKEL, E.; GASSMANN, O.; CHESBROUGH, H. Open R&D and Open Innovation: Exploring the phenomenon. **R&D Management**, v. 39, n. 4, p. 311–316, 2009. Disponível em: <https://doi.org/10.1111/j.1467-9310.2009.00570.x>. Acesso em: 28 jun. 2024.

GACEK, C.; ARIEF, B. The many meanings of open source. **IEEE software**, v. 21, n. 1, p. 34–40, 2004. Disponível em: <https://doi.org/10.1109/MS.2004.1259206>. Acesso em: 28 jun. 2024.

GAIMON, C.; SINGHAL, V. Flexibility and the Choice of Manufacturing Facilities Under Short Product Life Cycles. **European journal of operational research**, v. 60, n. 2, p. 211–223, 1992. Disponível em: <https://doi.org/10.1016/0377-2217(92)90094-P>. Acesso em: 28 jun. 2024.

GALLAGHER, J.; WEST, S. Patterns of Open Innovation in Open Source Software. Em: CHESBROUGH, H.; VANHAVERBEKE, W.; WEST, J. (Eds.). **Open Innovation: Researching a New Paradigm**. 1. ed. Nova York: Oxford University Press, 2006. p. 82–106. Disponível em: <https://doi.org/10.1093/oso/9780199290727.003.0005>. Acesso em: 28 jun. 2024.

GASSMANN, O.; ENKEL, E.; CHESBROUGH, H. The Future of Open Innovation. **R&D Management**, v. 40, n. 3, p. 213–221, 2010. Disponível em: <https://doi.org/10.1111/j.1467-9310.2010.00605.x>. Acesso em: 28 jun. 2024.

GEROSA, M.; WIESE, I.; TRINKENREICH, B.; LINK, G.; ROBLES, G.; TREUDE, C.; STEINMACHER, I.; SARMA, A. **The Shifting Sands of Motivation: Revisiting what drives contributors in open source**. *In*: IEEE/ACM 43RD INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING, 43., 2021. **Anais eletrônicos** [...] Madrid: IEEE, 2021, p. 1046-1058. Disponível em: <https://doi.org/10.1109/icse43902.2021.00098>. Acesso em: 28 jun. 2024.

GOLDMAN, R.; GABRIEL, R. P. **Innovation Happens Elsewhere: Open source as Business Strategy**. 1. ed. São Francisco: Morgan Kaufmann, 2005. 427 p. ISBN: 9786611078195.

GOMPERS, P.; LERNER, J. Short-term America Revisited? Boom and Bust in the Venture Capital Industry and the Impact on Innovation. **Innovation policy and the economy**, v. 3, p. 1–27, 2003. Disponível em: <https://doi.org/10.1086/ipe.3.25056151>. Acesso em: 28 jun. 2024.

GRAND, S.; VON KROGH, G.; LEONARD, D.; SWAP, W. Resource Allocation Beyond Firm Boundaries. **Long Range Planning**, v. 37, n. 6, p. 591–610, 2004. Disponível em: <https://doi.org/10.1016/j.lrp.2004.09.006>. Acesso em: 28 jun. 2024.

HECKER, F. **Writings**. Disponível em: <https://frankhecker.com/writings/>. Acesso em: 19 jun. 2024.

HENKEL, J. Selective Revealing in Open Innovation Processes: The case of embedded Linux. **Research Policy**, v. 35, n. 7, p. 953–969, 2006. Disponível em: <https://doi.org/10.1016/j.respol.2006.04.010>. Acesso em: 28 jun. 2024.

HIPPEL, E. VON; KROGH, G. VON. Open Source Software and the “Private-collective” Innovation Model: Issues for organization science. **Organization Science**, v. 14, n. 2, p. 209–223, 2003. Disponível em: <https://doi.org/10.1287/orsc.14.2.209.14992>. Acesso em: 28 jun. 2024.

HOLTGRAVE, U.; WERLE, R. De-Commodifying Software? Open Source Software Between Business Strategy and Social Movement. **Science & Technology Studies**, v. 14, n. 2, p. 43–65, 2001. Disponível em: <https://doi.org/10.23987/sts.55135>. Acesso em: 28 jun. 2024.

INTERNATIONAL COOPERATIVE ALLIANCE (ICA). **Cooperative identity, values & principles**. Disponível em: <https://ica.coop/en/cooperatives/cooperative-identity>. Acesso em: 17 jun. 2024.

JIANG, L.; EBERLEIN, A. **An Analysis of the History of Classical Software Development and Agile Development**. *In*: 2009 IEEE International Conference on Systems, Man and Cybernetics, 2009, San Antonio. **Anais eletrônicos** [...] San Antonio: IEEE, 2009. Disponível em: <https://doi.org/10.1109/ICSMC.2009.5346888>. Acesso em: 28 jun. 2024.

KAISER, U.; KONGSTED, H. C.; RØNDE, T. Does the Mobility of R&D Labor Increase Innovation? **Journal of Economic Behavior & Organization**, v. 110, p. 91–105, 2015. Disponível em: <https://doi.org/10.1016/j.jebo.2014.12.012>. Acesso em: 28 jun. 2024.

KAISLER, S. Lisp: An AI programming language. **IEEE Journal of Oceanic Engineering**, v. 11, n. 4, p. 468–473, 1986. Disponível em: <https://doi.org/10.1109/JOE.1986.1145204>. Acesso em: 28 jun. 2024.

KARELS, M. J. Commercializing Open Source Software: Many have tried, a few are succeeding, but challenges abound. **ACM Queue: Tomorrow's Computing Today**, v. 1, n. 5, p. 46–55, 2003. Disponível em: <https://doi.org/10.1145/945074.945125>. Acesso em: 28 jun. 2024.

KERNER, S. M. Linux Foundation Launches Node.js Foundation. **eWEEK**, [s.l.], 17 jun. 2015. Disponível em: <https://www.eweek.com/development/linux-foundation-launches-node.js-foundation/>. Acesso em: 10 set. 2023.

LAKKA, S.; VAROUTAS, D.; MARTAKOS, D. Impact of OSS on Software Markets - An Evaluation. In: MEDITERRANEAN CONFERENCE ON INFORMATION SYSTEMS 2009 PROCEEDINGS, 4, 2009, Atenas. **Anais eletrônicos** [...] Atenas: [s.n], 2009. p. 588-599. Disponível em: <https://aisel.aisnet.org/mcis2009/56>. Acesso em: 9 set. 2023.

LARDINOIS, F. Microsoft Launches .NET Foundation To Foster The .NET Open Source Ecosystem. **TechCrunch**, [s.l.], 3 abr. 2014. Disponível em: <https://techcrunch.com/2014/04/03/microsoft-launches-net-foundation-to-foster-the-net-open-source-ecosystem/>. Acesso em: 10 set. 2023.

LAURSEN, K.; SALTER, A. Open for Innovation: The role of openness in explaining innovation performance among U.K. manufacturing firms. **Strategic management journal**, v. 27, n. 2, p. 131–150, 2006. Disponível em: <https://doi.org/10.1002/smj.507>. Acesso em: 28 jun. 2024.

LEMOS, R. **Direito, Tecnologia e Cultura**. 1. ed. Rio de Janeiro: FGV Editora, 2005. 212 p. ISBN: 9788522505166.

LERNER, J.; TIROLE, J. Some simple economics of open source. **The Journal of Industrial Economics**, v. 50, n. 2, p. 197–234, 2002. Disponível em: <https://doi.org/10.1111/1467-6451.00174>. Acesso em: 28 jun. 2024.

LEVY, S. **Hackers: Heroes of the computer revolution - 25th anniversary edition**. 1. ed. Sebastopol: O'Reilly Media, 2010. ISBN: 9781449388393.

LIN, Y. Hybrid innovation: The dynamics of collaboration between the FLOSS community and corporations. **Knowledge Technology & Policy**, v. 18, n. 4, p. 86–100, 2006. Disponível em: <https://doi.org/10.1007/s12130-006-1005-7>. Acesso em: 28 jun. 2024.

MARQUES, I. da C.; FILHO, R. A. M. de S. Fazendo-medindo a Economia do Software: Microsoft versus open source - dos primeiros encontros até 2005. **Redes**, Buenos Aires, v. 14,

n. 27, p. 141-162, maio 2008. Disponível em: <https://www.redalyc.org/articulo.oa?id=90717063006>. Acesso em: 9 set. 2023.

MATOS, L. B. de S.; TAVARES, L. E.; FERREIRA, L. de M. Mecanismos de Apropriabilidade do Software: um Estudo sobre o Porto Digital. *In*: XIV CONGRESSO ALTEC, 14., 2011, Lima. **Anais eletrônicos** [...] Fortaleza: Universidade Estadual do Ceará, 2011. p. 1-16. Disponível em: <https://hal.science/hal-03088383>. Acesso em: 28 jun. 2024.

MCKUSICK, M. K. Twenty Years of Berkeley Unix: From AT&T-owned to freely redistributable. Em: DIBONA, C.; OCKMAN, S.; STONE, M. (Eds.). **Open Sources: Voices from the Open Source Revolution**. 1. ed. Sebastopol: O'Reilly & Associates, 1999. p. 23–24. ISBN: 9781565925823.

MEDAPPA, P. K.; SRIVASTAVA, S. C. Ideological Shifts in Open Source Orchestration: Examining the influence of licence choice and organisational participation on open source project outcomes. **European Journal of Information Systems**, v. 29, n. 5, p. 500–520, 2020. Disponível em: <https://doi.org/10.1080/0960085x.2020.1756003>. Acesso em: 28 jun. 2024.

MEDEIROS, H. G. Propriedade Intelectual na Sociedade Informacional: Produção e proteção de bens imateriais em tempos de capitalismo cognitivo. In: Encontro Nacional do CONPEDI, 23., 2014, Florianópolis. **Anais eletrônicos** [...] Florianópolis: CONPEDI, 2014. p. 474-492. ISBN: 9788568147122. Disponível em: <http://publicadireito.com.br/publicacao/ufsc/livro.php?gt=122>. Acesso em: 28 jun. 2024.

MICROSOFT. Microsoft Contributes Linux Drivers to Linux Community. **Microsoft PressPass**, Redmond, 20 jul. 2009. Disponível em: <https://web.archive.org/web/20120418154719/http://www.microsoft.com/en-us/news/features/2009/jul09/07-20linuxqa.aspx>. Acesso em: 9 set. 2023.

MØEN, J. Is Mobility of Technical Personnel a Source of R&D Spillovers? **Journal of labor economics**, v. 23, n. 1, p. 81–114, 2005. Disponível em: <https://doi.org/10.1086/425434>. Acesso em: 28 jun. 2024.

MONGODB INC. MongoDB issues new server side public license for MongoDB community server, **MongoDB**, 16 out. 2018. Disponível em: <https://www.mongodb.com/company/newsroom/press-releases/mongodb-issues-new-server-side-public-license-for-mongodb-community-server>. Acesso em: 28 jun. 2024.

MORGAN, L.; FINNEGAN, P. Open Innovation in Secondary Software Firms: An exploration of managers' perceptions of open source software. **ACM SIGMIS Database: the DATABASE for Advances in Information Systems**, v. 41, n. 1, p. 76–95, 2010. Disponível em: <https://doi.org/10.1145/1719051.1719056>. Acesso em: 28 jun. 2024.

NOFRE, D.; PRIESTLEY, M.; ALBERTS, G. When Technology Became Language: The origins of the linguistic conception of computer programming, 1950–1960. **Technology and Culture**, v. 55, n. 1, p. 40–75, 2014. Disponível em: <https://doi.org/10.1353/tech.2014.0031>. Acesso em: 28 jun. 2024.

O'HARA, M. T.; WATSON, R. T.; KAVAN, C. B. Managing the Three Levels of Change. **Information Systems Management**, v. 16, n. 3, p. 63–70, 1999. Disponível em: <https://doi.org/10.1201/1078/43197.16.3.19990601/31317.9>. Acesso em: 28 jun. 2024.

O'MAHONY, S. The Governance of Open Source Initiatives: What does it mean to be community managed? **Journal of Management & Governance**, v. 11, n. 2, p. 139–150, 2007. Disponível em: <https://doi.org/10.1007/s10997-007-9024-7>. Acesso em: 28 jun. 2024.

OPEN SOURCE INITIATIVE (OSI). The Open Source Definition. **Open Source Initiative**, 22 mar. 2007. Disponível em: <https://opensource.org/osd>. Acesso em: 20 jun. 2024.

OSHRI, I.; DE VRIES, H. **Standards Battles in Open Source Software: The Case of Firefox**. Basingstoke: Palgrave Macmillan, 2008. 203 p. ISBN: 9780230220720.

PERENS, B. The Open Source Definition. Em: DIBONA, C.; OCKMAN, S.; STONE, M. (Eds.). **Open Sources: Voices from the Open Source Revolution**. 1. ed. Sebastopol: O'Reilly & Associates, 1999. p. 79–86. ISBN: 9781565925823.

PERR, J.; APPLEYARD, M. M.; SULLIVAN, P. Open for Business: Emerging business models in open source software. **International Journal of Technology Management**, v. 52, n. 3/4, p. 432, 2010. Disponível em: <https://doi.org/10.1504/IJTM.2010.035984>. Acesso em: 28 jun. 2024.

PICCIOTTO, S.; CAMPBELL, D. Whose Molecule is it Anyway? Private and social perspectives on intellectual property. Em: HUDSON, A. (Ed.). **New perspectives on property law, obligations and restitution**. Londres: Routledge-Cavendish, 2004. p. 279–304. Disponível em: <https://doi.org/10.4324/9781843145714. ISBN: 9781843145714>. Acesso em: 28 jun. 2024.

QUARTERMAN, J. S.; WILHELM, S. **UNIX®, POSIX, and Open Systems: The open standards puzzle**. 1. ed. Boston: Addison Wesley, 1993. 448 p. ISBN: 9780201527728.

QUITTNER, J.; SLATALLA, M. **Speeding the Net: the inside story of Netscape & how it challenged Microsoft**. 1. ed. Londres: Orion Business, 1999. 323 p. ISBN: 9780752813776.

RADITS, M. **A Business Ecology Perspective on Community-Driven Open Source**: The case of the free and open source content management system Joomla. Linköping: Linköping University Electronic Press, 2019. Disponível em: <https://www.diva-portal.org/smash/get/diva2:1265888/FULLTEXT03.pdf>. Acesso em: 10 set. 2023.

RAYMOND, E. S. **The Cathedral and the Bazaar: Musings on Linux and open source by an accidental revolutionary**. 2. ed. Sebastopol: O'Reilly Media, 2001. ISBN: 9780596001087.

RAYMOND, E. S. The Revenge of the Hackers. Em: DIBONA, C.; OCKMAN, S.; STONE, M. (Eds.). **Open Sources: Voices from the open source revolution**. 1. ed. Sebastopol: O'Reilly & Associates, 1999. p. 96–101. ISBN: 9781565925823.

RITCHIE, D. M. The Evolution of the Unix Time-sharing System. **AT&T Bell Laboratories Technical Journal**, v. 63, n. 8, p. 1577–1593, 1984. Disponível em: <https://doi.org/10.1002/j.1538-7305.1984.tb00054.x>. Acesso em: 28 jun. 2024.

RITCHIE, D. M.; THOMPSON, K. The UNIX time-sharing system. **Communications of the ACM**, v. 17, n. 7, p. 365–375, 1974. Disponível em: <https://doi.org/10.1145/361011.361061>. Acesso em: 28 jun. 2024.

ROMIJN, H.; ALBALADEJO, M. Determinants of innovation capability in small electronics and software firms in southeast England. **Research Policy**, v. 31, n. 7, p. 1053–1067, 2002. Disponível em: <https://doi.org/10.1016/S0048-7333(01)00176-7>. Acesso em: 28 jun. 2024.

RUTHERFORD, M. The CLOUD Act: Creating executive branch monopoly over cross-border data access. **Berkeley Technology Law Journal**, v. 34, n. 4, p. 1177–1204, 2019. Disponível em: <https://doi.org/10.15779/Z387940V34>. Acesso em: 28 jun. 2024.

SAEBI, T.; FOSS, N. J. Business Models for Open Innovation: Matching heterogeneous open innovation strategies with business model dimensions. **European Management Journal**, v. 33, n. 3, p. 201–213, 2015. Disponível em: <https://doi.org/10.1016/j.emj.2014.11.002>. Acesso em: 28 jun. 2024.

SAFARI, H.; SABRI, N.; SHAHSAVAN, F.; BAHRAK, B. An Analysis of GitLab's Users and Projects Networks. *In*: 10TH INTERNATIONAL SYMPOSIUM ON TELECOMMUNICATIONS, 10., 2020, Teerã. **Anais eletrônicos** [...] Teerã: IEEE, 2020, p. 194-200. Disponível em: <https://doi.org/10.1109/IST50524.2020.9345844>. Acesso em: 28 jun. 2024.

SALUS, P. H. **A Quarter Century of UNIX**. 1. ed. Boston: Addison Wesley, 1994. 272 p. ISBN: 9780201547771.

SCHUMPETER, J. A. **Teoria do Desenvolvimento Econômico: Uma investigação sobre lucros, capital, crédito, juro e o ciclo econômico**. Tradução: Maria Sílvia Possas. São Paulo: Nova Cultural, 1997. 242 p. ISBN: 9788535109153.

SCHWARTZ, J. In Major Shift, Microsoft Contributes Code to Linux Community. **Visual Studio Magazine**, [s.l.], 21 jul. 2009. News. Disponível em: <https://web.archive.org/web/20090725202523/https://visualstudiomagazine.com/articles/2009/07/21/microsoft-contributes-code-to-linux-community.aspx>. Acesso em: 9 set. 2023.

SHANE, S. **Academic Entrepreneurship: University spinoffs and wealth creation**. Cheltenham: Edward Elgar Publishing, 2004. 352 p. ISBN: 9781843764540. Disponível em: <https://doi.org/10.4337/9781843769828>. Acesso em: 28 jun. 2024.

SMIT, T. The appropriability regime as a tool to measure knowledge protection. In: INTERNATIONAL BUSINESS ADMINISTRATION BACHELOR THESIS CONFERENCE, 3., 2014, Enschede. **Anais eletrônicos** [...] Enschede: University of Twente, 2014, p. 1-12. Disponível em: <https://purl.utwente.nl/essays/65371>. Acesso em: 28 jun. 2024.

SOARES, M. D. S. Metodologias Ágeis Extreme Programming e Scrum para o Desenvolvimento de Software. **Revista Eletrônica de Sistemas de Informação**, v. 3, n. 1, 2004. Disponível em: <https://doi.org/10.21529/resi.2004.0301006>. Acesso em: 28 jun. 2024.

SOFTWARE PUBLISHERS ASSOCIATION. Competition in the Network Market: The Microsoft Challenge. **Software Publishers Association**, jun. 1998. Disponível em: <https://www.digiater.nl/openvms/decus/vmslt98a/net98a/ms-anticompetitive-practices.pdf>. Acesso em: 7 set. 2023.

SOUZA JÚNIOR, R. R. de. Soberano oculto: análise do papel do Estado norte-americano no fomento aos setores industriais ligados à computação. **Revista de Ciências Humanas**, v. 54, p. 1–15, 2021. Disponível em: <https://doi.org/10.5007/2178-4582.2020.e68260>. Acesso em: 28 jun. 2024.

STALLMAN, R. M. **Free Software, Free Society: Selected essays of Richard M. Stallman**. Boston: Free Software Foundation, 2015. 293 p. ISBN: 9780983159254.

STALLMAN, R. M. The GNU Manifesto. **Dr. Dobb's Journal**, v. 101, n. 14, p. 30–31, mar. 1985. Disponível em: <https://archive.org/details/1985-03-dr-dobbs-journal>. Acesso em: 28 jun. 2024.

STEINMUELLER, W. E. **The U.S. Software Industry: An analysis and interpretative history**. University of Maastricht, 1995. 57 p. Disponível em: <https://doi.org/10.26481/umamer.1995006>. Acesso em: 28 jun. 2024.

STEWART, K.; KHAN, S.; GERMAN, D. 2020 Linux Kernel History Report. **The Linux Foundation**, ago. 2020. Disponível em: <https://www.linuxfoundation.org/resources/publications/linux-kernel-history-report-2020>. Acesso em: 5 set. 2023.

TAKAHASHI, N.; TAKAMATSU, T. UNIX license makes Linux the last major piece of the puzzle. **Annals of Business Administrative Science**, v. 12, n. 3, p. 123–137, 2013. Disponível em: <https://doi.org/10.7880/abas.12.123>. Acesso em: 28 jun. 2024.

TEECE, D. J. Profiting from Technological Innovation: Implications for integration, collaboration, licensing and public policy. **Research Policy**, v. 15, n. 6, p. 285–305, 1986. Disponível em: <https://doi.org/10.1016/0048-7333(86)90027-2>. Acesso em: 28 jun. 2024.

THOMAS, H. The History of Unix in the History of Software. *In*: **La recherche sur les systèmes: des pivots dans l'histoire de l'informatique – II/II**, 1., 2017, [s.l.]. **Anais eletrônicos** [...] Paris: Conservatoire national des arts et métiers, 2017. p. 77-90. Disponível em: <https://hal.science/hal-03027081>. Acesso em: 28 jun. 2024.

TIDD, J.; BESSANT, J.; PAVITT, K. **Managing Innovation: Integrating technological, market and organizational change**. 3. ed. Chichester: John Wiley & Sons, 2005. ISBN: 9780470093269.

VALLOPPILLIL, V; COHEN, J. **Linux OS Competitive Analysis: The next Java-VM?**. Microsoft: [s.l.], p. 1-30, 11 ago. 1998. Disponível em:

<http://www.groklaw.net/pdf/iowa/www.iowaconsumercase.org/011607/6000/PX06464.pdf>. Acesso em: 7 set. 2023.

VALLOPPILLIL, V. **Open Source Software: A (New?) Development Methodology**. Microsoft: [s.l.], p. 1-32, 11 ago. 1998. Disponível em: <http://www.groklaw.net/pdf/iowa/www.iowaconsumercase.org/011607/6000/PX06501.pdf>. Acesso em: 7 set. 2023.

VAUGHAN-NICHOLS, S. Elastic changes open-source license to monetize cloud-service use, **ZDNET**, 20 jan. 2021. Disponível em: <https://www.zdnet.com/article/elastic-changes-open-source-license-to-monetize-cloud-service-use/>. Acesso em: 28 jun. 2024.

VESEY, J. T. Time-to-market: Put speed in product development. **Industrial marketing management**, v. 21, n. 2, p. 151–158, 1992. Disponível em: <https://doi.org/10.1016/0019-8501(92)90010-Q>. Acesso em: 28 jun. 2024.

WATZINGER, M.; FACKLER, T. A.; NAGLER, M.; SCHNITZER, M.;. How Antitrust Enforcement Can Spur Innovation: Bell Labs and the 1956 consent decree. **American Economic Journal. Economic Policy**, v. 12, n. 4, p. 328–359, 2020. Disponível em: <https://doi.org/10.1257/pol.20190086>. Acesso em: 28 jun. 2024.

WEST, J.; GALLAGHER, S. Challenges of Open Innovation: The paradox of firm investment in open-Source software. **R&D Management**, Oxford, v. 36, n. 3, p. 319-331, 2006. Disponível em: <https://doi.org/10.1111/j.1467-9310.2006.00436.x>. Acesso em: 7 set. 2023.

WILLIAMS, S. **Free as in Freedom (2.0): Richard Stallman and the free software revolution**. Boston: Free Software Foundation, 2010. ISBN: 9780983159216.

WU, T. Tech Dominance and the Policeman at the Elbow. Em: WERBACH, K. (Ed.). **After the Digital Tornado: Networks, algorithms, humanity**. Cambridge: Cambridge University Press, 2020. p. 89–90. ISBN: 978-1-108-61001-8. Disponível em: <https://doi.org/10.1017/9781108610018>. Acesso em: 28 jun. 2024.

YIN, R. K. **Estudo de Caso: Planejamento e métodos**. Tradução: Daniel Grassi. 2. ed. Porto Alegre: Bookman, 2001. ISBN: 9788573078527.

ZERBE, R. O.; MCCURDY, H. E. The failure of market failure. **Journal of Policy Analysis and Management**, v. 18, n. 4, p. 558–578, 1999. Disponível em: <https://doi.org/10.1002/(SICI)1520-6688(199923)18:4<558::AID-PAM2>3.0.CO;2-U>. Acesso em: 28 jun. 2024.

ZOLKIFLI, N. N.; NGAH, A.; DERAMAN, A. Version Control System: A review. **Procedia Computer Science**, v. 135, p. 408–415, 2018. Disponível em: <https://doi.org/10.1016/j.procs.2018.08.191>. Acesso em: 28 jun. 2024.