UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE ARTES E COMUNICAÇÃO
CURSO DE ARQUITETURA E URBANISMO

RODRIGO COSTA TEIXEIRA

ARQUITETURA EM CÓDIGO ABERTO

CRÍTICA AO CONCEITO E ALGUMAS PERSPECTIVAS

RECIFE
2022

RODRIGO COSTA TEIXEIRA

ARQUITETURA EM CÓDIGO ABERTO

CRÍTICA AO CONCEITO E ALGUMAS PERSPECTIVAS

Trabalho de Conclusão de Curso apresentado à Coordenação de Graduação em Arquitetura e Urbanismo da Universidade Federal de Pernambuco para a obtenção do grau de Bacharel em Arquitetura urbanismo.

Orientador: Pascal Machado

RECIFE

2022

Ficha de identificação da obra elaborada pelo autor,
através do programa de geração automática do SIB/UFPE

Teixeira, Rodrigo Costa.
Arquitetura em Código Aberto: crítica ao conceito e algumas perspectivas / Rodrigo Costa Teixeira. - Recife, 2022.
77 : il.

Orientador(a): Pascal Machado
Trabalho de Conclusão de Curso (Graduação) - Universidade Federal de Pernambuco, Centro de Artes e Comunicação, Arquitetura e Urbanismo - Bacharelado, 2022.

1. código aberto. 2. tecnologias digitais. 3. metodologia de projeto. 4. movimento social. I. Machado, Pascal. (Orientação). II. Título.

720 CDD (22.ed.)

AGRADECIMENTOS

Agradeço meus pais, Marcus e Monica, por terem torcido mais do que ninguém para que eu entregasse um trabalho de qualidade. Com muita sensibilidade, me incentivaram e me cobraram. E graças a educação que recebi, estou aqui.

Agradeço Rafael, meu irmão, que, mesmo atolado de coisa da faculdade, sempre demonstrou interesse na minha pesquisa. No início era difícil resumir o tema, mas rolou.

Agradeço também a Vitória que nos momentos mais chatinhos ela me confortou, me deu forças e também me divertiu. Que bom que foi assim.

Agradeço ao meu orientador, Pascal Machado por ter acreditado e se empolgado com a ideia desde o princípio. Sua paciência, pois demorei para conseguir delimitar o tema da minha pesquisa.

Agradeço os amigos que dividiram tantos momentos comigo. Muitos me acolheram quando eu estava desestimulado com o curso e isso me fez seguir. As palavras e o compartilhamento de experiências foram fundamentais.

Por fim, agradeço à Deus. Mesmo que não possa compreendê-lo, tenho um sentimento muito sincero de gratidão.

SUMÁRIO

INTRODUÇÃO

O conceito de software livre e de código aberto é amplamente difundido no campo da informática, sendo um movimento social e uma metodologia de desenvolvimento de programas que rompe com o modelo tradicional e hierarquizado de grandes empresas, promovendo uma alternativa não só mais eficiente, como também, para muitos defensores, moralmente correta. Diante das mudanças tecnológicas, culturais e econômicas que caracterizam o que se convém chamar de Era da Informação, bem como com a força dos ideais defendidos por esse movimento, o discurso do código aberto extrapolou a ciência da computação, influenciando outras áreas, como design, direito, arte, gestão de empresas, urbanismo, e também arquitetura.

O compartilhamento dos códigos-fonte era comum desde o surgimento dos primeiros computadores, especialmente no meio universitário. O código circulava entre as equipes dos laboratórios de pesquisa, de modo que soluções desenvolvidas em uma universidade eram frequentemente usadas e melhoradas em outras, formando-se uma rede de desenvolvimento tecnológico espontânea e colaborativa. A prática, porém, foi perdendo espaço para uma crescente indústria de software que, ao invés de fornecer o código-fonte aos seus usuários, tornava-o secreto. E, para maximizar os lucros, as empresas do ramo restringiam, por meio das leis de copyrights, o compartilhamento entre os usuários.

Esse modelo rapidamente tornou-se hegemônico, mas provocou a revolta de muitos pesquisadores e entusiastas, que perceberam o ambiente de troca de conhecimento e de tecnologia do qual faziam parte cedendo para um modelo pautado na competitividade, na restrição de uso e no monopólio da informação. Como forma de resistência, buscando resgatar o ambiente colaborativo de outrora, Richard Stallman, pesquisador do Laboratório de Inteligência Artificial do MIT, anunciou em 1983 o Projeto GNU, um novo sistema operacional, em código aberto,

desenvolvido colaborativamente, cujo compartilhamento não só era permitido, como também incentivado. Nascia ali o movimento software livre.

A iniciativa atraiu muitos desenvolvedores que o ajudaram escrevendo códigos e reportando erros, estabelecendo em torno do Projeto GNU uma comunidade de programadores e usuários. Anos depois, juntando forças com outro projeto em código aberto, o Linux, desenvolvido inicialmente pelo finlandês Linus Torvalds, o sistema GNU/Linux, ou apenas Linux, tornou-se o maior projeto de software livre do mundo. O sucesso do Linux e do seu modelo de desenvolvimento surpreendeu Eric Raymond, hoje considerado um dos principais porta-vozes do código aberto no mundo. No documentário *Revolution OS* (2001), ele diz:

> Eu era, na verdade, um dos primeiros programadores do GNU. E eu estava absolutamente surpreso! Porque eu tinha sido engenheiro de software por quase 15 anos até aquela época. E de acordo com todas as regras que eu conhecia (...), o Linux deveria ser um desastre, mas não era. Ao invés disso, era algo maravilhoso, e eu estava determinado a descobrir como eles estavam encaminhando aquilo. (REVOLUTION OS, 2001, tradução nossa)

No campo da arquitetura, o workshop intitulado *Open Source Media Architecture*, organizado pelo RIXC, realizado em maio de 2004 em Riga, capital da Letônia, foi um marco na discussão sobre definições e diretrizes para uma arquitetura em código aberto. Desde então, outros trabalhos foram publicados, como o *Urban Versioning System 1.0* (FULLER; HAQUE, 2008), por exemplo, buscando no movimento respostas para temáticas recorrentes do campo da arquitetura, como o direito a moradia de qualidade, a inserção dos moradores nos processos de concepção e produção do espaço, e também o papel do arquiteto em um mundo permeado e interfaceado pelas tecnologias digitais.

Em 2011, a revista Domus convidou o arquiteto Carlo Ratti para que ele trouxesse sua visão sobre o tema. A ideia inicial era que ele produzisse um artigo

para a revista, mas para ser coerente ele preferiu escrever colaborativamente com outros arquitetos e interessados. Para isso, foi criada uma página na Wikipedia e ela foi escrita ali mesmo por dezenas de autores em diversas partes do mundo. O texto foi publicado na revista, mas a página na Wikipedia se mantém ativa até hoje, aberta a todos, não só para leitura, mas também para edição. Trata-se de um documento vivo, aperfeiçoado e complementado ao longo do tempo. Como forma de ampliar o debate, quatro anos depois foi publicado o livro *Open Source Architecture* (CLAUDEL; RATTI 2015), escrito também em um modelo de coautoria com outros arquitetos renomados, tornando-se a principal referência sobre o assunto. Segundo o livro e manifesto:

> A arquitetura de código aberto é um paradigma emergente que defende novas formas de imaginar e desenvolver espaços reais ou virtuais em uma infraestrutura universal. Partindo de referências tão diversas, como cultura do código aberto, design modular, arquitetura de vanguarda, ficção científica, teoria da linguagem e neurocirurgia, ela busca uma abordagem inclusiva do design espacial, uma apropriação colaborativa do processo e das ferramentas de design, por profissionais e cidadãos comuns. (CLAUDEL; RATTI, 2015, pág. 131, tradução nossa)

Além das discussões teóricas, muitas iniciativas têm se definido como *open source*. Tem-se, por exemplo, escritórios como o Opening Design, com um discurso de transparência e interdisciplinaridade, e o ELEMENTAL, de Alejandro Aravena, que disponibiliza alguns projetos na intetnet. Existem também sistemas construtivos, como a Wikihouse, pautada nas tecnologias de fabricação digital, no compartilhamento de informação e na construção colaborativa, mas também a extinta Paperhouses, ou ainda o Bricksource, que consiste em um banco de dados online de padrões para alvenaria paramétrica, entre outras.

Sem dúvida, trata-se de uma metodologia e um movimento social bem sucedido no campo da informática, mas apesar da multiplicidade de iniciativas na arquitetura ainda não há clareza sobre o que, de fato, possa ser considerado uma arquitetura em código aberto. As abordagens teóricas ainda são muito amplas e,

por isso, imprecisas, sem uma aplicação evidente. Do ponto de vista prático, o *open source* acaba sendo empregado, ou como sinônimo para processos participativos, ou para o simples ato de disponibilizar plantas e cortes online. Para um programador, é muito simples discernir o que é ou não um software em código aberto. Trata-se de um conceito trivial, mas quando se importa para a arquitetura, nota-se uma abstração e uma aplicação que não condiz com as potencialidades do modelo de produção presente no campo da informática.

Surgem então alguns questionamentos. Afinal, até que ponto a ideia de código aberto se justifica no campo da arquitetura? Existe nesse movimento alguma característica que o difira de outros conceitos já debatidos entre arquitetos? O movimento código aberto deve ser compreendido apenas como uma inspiração ou pode representar, de fato, um novo paradigma de concepção e produção do espaço? Quais seriam as características e particularidades de uma arquitetura em código aberto?

Dividida em três capítulos, a presente dissertação propõe um olhar mais crítico sobre esse conceito, buscando compreender as questões fundamentais que contribuíram para o surgimento e o sucesso do movimento código aberto, e de que forma essas questões, específicas do campo da informática, podem dialogar com o campo da arquitetura, especialmente em um momento histórico em que todos os campos produtivos estão cada vez mais dependentes das tecnologias digitais. Vale destacar que a pesquisa, a princípio, não trás um problema específico a ser resolvido pelo código aberto. Pelo contrário, objetiva-se traçar um panorama, explorando o conceito, compreendendo seu potencial e suas limitações, e apresentando possíveis diretrizes.

O primeiro capítulo, portanto, discute sobre o código aberto no campo da informática, tanto como uma metodologia de desenvolvimento tecnológico, como também um movimento social.

O segundo capítulo tece um raciocínio com base nas aproximações conceituais já realizadas, identificando os principais argumentos e problemáticas

solucionadas. O livro *Open Source Architecture* (CLAUDEL; RATTI, 2015) e o artigo *A Communism of ideas: Towards an open-source architectural practice* (KASPORI, 2003) são um norte, mas outras referências são consultadas.

O terceiro capítulo, por meio da análise de algumas iniciativas práticas, apresenta uma crítica aos principais argumentos desenvolvidos no segundo e aponta outras problemáticas, pouco enfatizadas por esses autores, mas que são lições históricas do próprio movimento software livre.

1 CÓDIGO ABERTO NA INFORMÁTICA

1.1 ORIGEM DO SOFTWARE LIVRE

O contexto social e tecnológico que culminou com o surgimento do movimento software livre já vinha se formando desde o início dos anos 1980, com a popularização dos computadores pessoais e o crescimento da indústria de software. Um evento específico, porém, ocorrido no Laboratório de Inteligência Artificial do MIT, tornou a questão mais evidente para Richard Stallman, fundador e principal ideólogo do movimento. Após fracassadas tentativas de corrigir o mal funcionamento de uma impressora recém adquirida por sua universidade, Stallman decidiu contatar a Xerox, fabricante do equipamento, e solicitar o código-fonte do driver, software responsável por sua operação.

Código-fonte, ou simplesmente código, é um conjunto de instruções direcionadas a um computador escritas em uma linguagem de programação preestabelecida. Pode-se dizer que o código é a alma do software, é a sua essência. Ele está para o software, do mesmo modo que uma receita está para um bolo, ou um projeto arquitetônico está para um edifício. É a partir do código-fonte que o homem pode compreender o funcionamento de um software e, eventualmente, realizar alterações e aperfeiçoamentos. Sem ele, por outro lado, esse funcionamento torna-se uma mistério, apenas uma sequência confusa de zeros e uns. Quando se diz que um programa, um software, tem o código aberto, significa, em síntese, que o seu código-fonte está disponível na internet, podendo ser acessado, estudado, modificado e compartilhado por qualquer pessoa.

O código do driver da impressora foi rapidamente disponibilizado para a equipe de Stallman, que o corrigiu, e a impressora passou a operar normalmente. Esse mesmo problema vinha acometendo pesquisadores de outra universidade, mas quando souberam da solução encontrada no MIT, pediram o código com as correções, e foram atendidos. O problema de ambos estava resolvido.

Um ano depois, o MIT comprou novas e mais modernas impressoras da mesma empresa, as Xerox 9700, e elas apresentaram um problema semelhante. Dessa vez, porém, quando solicitado o código-fonte do driver, a Xerox impôs ao laboratório duas condições: que pagassem para obtê-lo e que se comprometessem a mantê-lo em segredo. Frustrado, Stallman recusou a oferta e buscou a universidade que eles haviam ajudado no passado. Ao contrário do MIT, essa universidade optou pelo pagamento e assinou o contrato de confidencialidade proposto. Se por um lado, conseguiram resolver o seu problema, por outro, não puderam compartilhar a solução, o que acabou criando um mal estar entre a comunidade universitária e uma pressão para que outras universidades e empresas adquirissem o código diretamente da Xerox.

Figura 1: Impressora Xerox 9700 em uma material publicitário.

Fonte: https://www.digibarn.com/collections/printers/xerox-9700/

Esse episódio ganhou relevância histórica porque tornou claro para Stallman uma tendência preocupante: a medida em que esse novo modelo de negócios crescia e alargava a margem de lucro da emergente indústria, ela também sufocava uma comunidade de universitários e hackers que tinham como prática essencial o compartilhamento e a produção coletiva de informação e tecnologia.

Para usufruir de qualquer computador moderno da época, início da década de 1980, não haviam alternativas senão adquirir um dos sistemas operacionais proprietários. O mais popular nas universidades era o Unix, desenvolvido inicialmente no Bell Labs da AT&T e adotado por hackers e acadêmicos porque, dentre outros motivos, não haviam, no princípio, restrições quanto a circulação do seu código. Mas após a uma mudança de gestão, a AT&T passou a processar universidades que o compartilhassem, fazendo uso das leis de copyrights,alegando serem donos daqueles códigos. Sobre o Unix, em entrevista ao documentário *Revolution OS* (2001), Stallman explicou:

> Os desenvolvedores desses sistemas não os compartilhavam com outras pessoas. Ao invés, eles tentaram controlar os usuários, dominar os usuários, restringi-los. Diziam: “se você pegar o sistema, você tem que assinar uma promessa que não vai compartilhar com ninguém”. E para mim isso era essencialmente uma promessa para ser uma má pessoa, para trair o resto do mundo, me retirar da sociedade, de uma comunidade cooperativa. E eu já havia experimentado o que acontece quando outras pessoas fazem isso conosco, quando eles se recusam a compartilhar conosco. Porque eles haviam assinado esses contratos. E isso feriu todo o laboratório, nos impedindo de fazer as coisas úteis que fazíamos antes. (REVOLUTION OS, 2001, tradução nossa)

Em resistência, Stallman decidiu escrever o GNU, um novo sistema operacional cujo compartilhamento, ao invés de restrito, seria incentivado. Sobre o nome, GNU, trata-se de um acrônimo recursivo que significa *GNU’s Not Unix*. Esse tipo de abreviatura era uma tradição entre os hackers, em especial os do MIT, evidenciando desde o início a intenção de resgatar elementos culturais do grupo.

Mas afirmar que o GNU não é Unix, além do explícito, quer dizer também que o GNU seria projetado para ser totalmente compatível com o Unix. Mesmo não sendo uma base ideal, Stallman optou por essa compatibilidade, ao invés de criar um sistema absolutamente do zero, para simplificar o seu desenvolvimento e viabilizar o trabalho colaborativo. Primeiro porque a maioria já estava acostumada com o Unix, e segundo porque ele pôde servir como um molde, de modo que cada programador, individualmente, pudesse desenvolver uma parte do GNU e essas partes seriam automaticamente compatíveis entre si.

Em 1983 ele anuncia o projeto ao público por meio do manifesto *Manifesto GNU*, lançando ali as bases ideológicas do movimento software livre. Em 1984, Stallman abre mão do seu cargo no MIT para se dedicar exclusivamente ao projeto, e em outubro de 1985 cria a Free Software Foundation, com o objetivo de dar suporte financeiro, jurídico, e estimular o uso e o desenvolvimento do GNU e de outros softwares livres. O conceito código aberto seria cunhado anos depois, após uma cisão no movimento software livre.

Entende-se como software livre, segundo a FSF, o software que compactua com as quatro liberdades:

- A liberdade de executar o programa como você desejar, para qualquer propósito (liberdade 0).
- A liberdade de estudar como o programa funciona, e adaptá-lo às suas necessidades (liberdade 1). Para tanto, acesso ao código-fonte é um pré-requisito.
- A liberdade de redistribuir cópias de modo que você possa ajudar outros (liberdade 2).
- A liberdade de distribuir cópias de suas versões modificadas a outros (liberdade 3). Desta forma, você pode dar a toda comunidade a chance de beneficiar de suas mudanças. Para tanto, acesso ao código-fonte é um pré-requisito.

O jeito mais simples de tornar um software livre seria disponibilizar o seu código na internet e colocá-lo em domínio público, abrindo mão de sua autoria e, consequentemente, de quaisquer direitos sobre ele. Essa condição jurídica, porém, incentivaria empresas a se apropriarem desses softwares, adicionando alguns poucos recursos, e vendendo-os como proprietários. O que até então era livre, estaria sendo executado em um software proprietário, perdendo assim as suas liberdades. Para evitar esse impasse, a Free Software Foundation criou a GNU General Public License (GPL), uma licença que se apropria das leis de direitos autorais, mas inverte sua lógica.

Ao invés fazer exigências que restringem o uso e o compartilhamento, como é o mais comum, ela define que o software poderá ser usado, copiado, modificado e redistribuído para quaisquer fins, desde que essas 4 liberdades sejam sempre garantidas no seu compartilhamento. Assim, caso uma empresa queira utilizar e modificar um software livre, ela só poderá redistribuí-lo com a mesma licença e, portanto, junto com o seu código-fonte. Pode-se dizer, portanto, que o software livre possui autores, mas não possui donos. Essa tática de subversão das copyrights passou a ser conhecida como *copyleft*, ou “deixe copiar”, em tradução livre, e foi inspiração para outras licenças, fora do campo da informática, como as Creative Commons, por exemplo.

Com boa adesão entre os hackers e a comunidade acadêmica em geral, em 1991 quase todos os componentes que juntos compreendiam o sistema Unix já haviam sido substituídos, exceto por um. Faltava ainda o núcleo central, ou *kernel*, considerado o cérebro do sistema operacional, responsável por gerenciar recursos e fazer a ponte entre o computador (hardware) e o restante dos programas (software). Esse atraso no desenvolvimento por parte da comunidade do GNU era ruim, por um lado, já que os usuários precisavam recorrer ao *kernel* proprietário do Unix, mas, por outro, o atraso criou o contexto para o surgimento do *kernel* Linux em agosto de 1991, o qual viria a ser anexado ao GNU, formando pela primeira vez

um sistema operacional completo e totalmente livre, o GNU/Linux, ou, mais popularmente, Linux, para a insatisfação da FSF.

Desenvolvido na Finlândia por Linus Torvalds, na época um estudante de ciência da computação pela Universidade de Helsinque, o Linux tinha um objetivo simples: fazer funcionar o novo computador pessoal do seu criador. Como disse Torvalds no anúncio do projeto, o novo *kernel* seria "apenas um hobby, nada grande ou profissional como o GNU" (1991). O Linux, assim como o GNU, era um software livre, porém por razões um pouco distintas: não havia um forte sentido ético por trás da iniciativa, muito menos o desejo de formar um movimento social em torno do seu projeto. Mas também não se tratava de um ato de caridade, como ele explica em sua autobiografia:

> Não estava apenas partilhando meu trabalho para que os outros pudessem achá-lo útil. Eu também queria feedback (tudo bem, e elogios). Não via razão para cobrar de pessoas que podiam de fato me ajudar a melhorar meu trabalho. (DIAMONDS; TORVALDS, 2001, p. 121, tradução nossa)

A comunidade de programadores e usuários do Linux se expandiu rapidamente. Milhares de voluntários ao redor do mundo foram acessando o código-fonte, corrigindo erros, adicionando recursos e melhorias, e construíram um novo sistema operacional, que logo nas suas primeiras versões, apresentou-se como uma alternativa mais flexível e robusta do que outros sistemas proprietários da época, como o MS-DOS e o Windows, ambos da Microsoft. Muitos outros projetos em código aberto foram surgindo, construindo um movimento de abrangência mundial, uma rede de desenvolvedores, unidos pela internet, mas também, e principalmente, por motivações pessoais semelhantes.

Buscando compreender o perfil dos indivíduos, programadores ou usuários, que se identificam e de alguma forma se inserem no movimento software livre, a socióloga Inês Pereira (2004), constatou três tipos motivações principais. Mesmo se tratando de um movimento tão diverso, que engloba tantas culturas e etnias,

essas motivações atuam como uma liga, dando ao movimento um senso de unidade e propósito.

A primeira motivação consiste no simples prazer em estudar e modificar o código-fonte dos programas. Essa experiência de aprendizado e exploração das próprias competências, bem como também do potencial tecnológico das máquinas, não é possível com o uso de softwares proprietários e de código fechado, que excluem do usuário a “lógica por trás”. Tal prazer corresponde a uma característica fundamental da cultura hacker, como observou o pesquisador finlandês Pekka Himanen (2001, pág. 18): "o primeiro valor a guiar a vida de um hacker é a paixão, ou seja, algum objetivo interessante que o move e que é de fato gerador de alegria em sua realização". No documentário *Revolution OS* (2001), Stallman comentou:

> Eu me juntei ao Laboratório de Inteligência Artificial do MIT em 1971. Me juntei a uma comunidade próspera de hackers, pessoas que amavam programar, amavam explorar o que eles podiam fazer com computadores. (...) Este era o meu trabalho, e eu o amava, todos nós amávamos. Esta era a razão de fazer aquilo. (REVOLUTION OS, 2001, tradução nossa)

O segundo motivo diz respeito aos princípios éticos e, especialmente, a ideia de liberdade, a qual Richard Stallman sempre valorizou. O software livre, sob esse ponto de vista, não é apenas o mais divertido, mas também a forma correta de usar a tecnologia, a que possibilita a democratização do conhecimento, promovendo uma sociedade mais justa e igualitária. Mas é importante lembrar que movimento software livre não surge apenas pelos valores que prega, mas também por uma conjuntura que se opunha a esses valores. O software livre é, acima de tudo, um ato de resistência a uma tendência de monopolização do conhecimento tecnológico e de sufocamento das comunidades colaborativas, que compreendiam a informação como um bem comum. Sem o aspecto nocivo da indústria de softwares, não haveria o conceito de software livre, muito menos um movimento social.

O terceiro motivo que costuma influenciar programadores e usuários a defenderem o uso de softwares livres relaciona-se a um desejo de independência, de autonomia frente às grandes empresas da indústria. Trata-se de uma retomada de controle sobre o software, não só por permitir que usuário investigue o código e identifique potenciais ameaças, mas também por evitar que decisões pautadas em dinâmicas de mercado venham afetar o usufruto desses softwares.

Mas além dos já citados, um quarto motivo, observado principalmente na comunidade do Linux, foi fundamental para popularização do software livre, atraindo não só indivíduos, mas também empresas bastante lucrativas para o movimento: a extrema eficiência do modelo de desenvolvimento baseado na abertura do código-fonte.

Muitos desenvolvedores e empresas perceberam que os softwares livres poderiam, não só ter muita qualidade, como poderiam ter também um menor custo de produção. Entretanto, o termo *free software* era incômodo, inibindo a adoção e o investimento. Primeiramente, o *free* é ambíguo, pois na língua inglesa ele pode indicar tanto a ideia de liberdade, quanto de gratuidade. Mesmo que praticamente todos os softwares livres, de fato, fossem gratuitos, esse não era o ponto central, visto que muitos outros softwares, de código fechado, são disponibilizados gratuitamente. Para evidenciar a diferença, Stallman costumava repetir em suas palestras que o *free* era como em *free speech, not free beer*. Apesar dessa distinção, o termo ainda não era muito bem visto, pois as empresas não queriam associar seus produtos a algo, teoricamente, sem valor. Outro ponto é que o discurso de liberdade, por si só, não interessava muito ao mercado, especialmente porque muitas empresas consideravam investir em software livres, porém também vendiam softwares em código fechado. Adotar esse discurso colocaria as empresas em uma situação desconfortável e até hipócrita.

Nesse contexto, surge o termo *open source*, comumente traduzido como código aberto, e a Open Source Iniciative. Na prática, softwares livres e softwares em código aberto são quase a mesma coisa. Mas enquanto a Free Software

Foundation tem como princípios os valores éticos e a defesa de que todo software deve ser aberto, a Open Source Iniciative (OSI) tem um viés mais pragmático, enfatizando a metodologia de trabalho e as estratégia de negócios, sem um maior envolvimento político ou filosófico. Ideologicamente, o GNU, por razões óbvias, está mais alinhado a FSF, enquanto o *kernel* Linux tem maior identificação com a OSI. Mas ao longo da presente pesquisa, software livre e software em código aberto poderão ser interpretados como sinônimos.

1.2 A CATEDRAL E O BAZAR

O que se observa no Linux, para além das suas qualidades técnicas, competindo desde as suas primeiras versões com poderosas empresas do ramo, como a Microsoft e a Apple, é a agilidade do seu processo de desenvolvimento, que propicia um ritmo frenético de correções e aprimoramentos, quase diários. Tal qualidade do Linux não advém da genialidade dos seus desenvolvedores, mas de sua natureza aberta. O software livre, de código aberto, surgiu por motivações éticas, mas logo demonstrou seu potencial enquanto nova e mais eficiente metodologia de produção.

Eric Raymond, hacker norte-americano, desenvolvedor do Unix por dez anos e um dos primeiros colaboradores do projeto GNU, mesmo com toda sua experiência em programação, surpreendeu-se com a eferverscência da comunidade do Linux. Em seu manifesto, *The Cathedral and the Bazar* (1998), ele distinguiu os dois estilos antagônicos de desenvolvimento:

> Eu acreditava que os softwares mais importantes (...) precisavam ser construídos como as catedrais, habilmente criados com cuidado por mágicos ou pequenos grupos de magos trabalhando em esplêndido isolamento (...). O estilo Linus Torvalds - libere cedo, libere frequentemente, delegue tudo que você puder, esteja aberto ao ponto da promiscuidade - veio como uma surpresa. Nenhuma catedral calma e respeitosa aqui - ao contrário, a comunidade Linux assemelhou-se a um

> grande e barulhento bazar de diferentes agendas e aproximações (...), de onde um sistema coerente e estável aparentemente só poderia emergir por uma sucessão de milagres. (RAYMOND, 1998, tradução nossa)

Em geral, a indústria de software tem uma política de desenvolvimento mais fechada e centralizadora. O software é compreendido como um produto que deve ser finalizado e, então, apresentado aos consumidores. Todo o seu desenvolvimento ocorre internamente às organizações, com o trabalho de programadores contratados. As decisões não partem dos próprios desenvolvedores, elas são tomadas de cima para baixo por presidentes ou CEO's, pautando-se sempre em estratégias de mercado e análises de risco. Esse fechamento, associado ao uso extensivo de patentes e copyrights, é o motor da inovação tecnológica dessa indústria, configurando um ambiente de sigilos e forte concorrência inter-empresarial (PEREIRA, 2004).

Em um ecossistema de código aberto, por outro lado, é possível, comum, e até desejável, que softwares copiem outros softwares. Ao invés de começar do zero, os desenvolvedores buscam algum código de outro software existente que possa servir de base para a sua iniciativa. No caso do GNU, não foi possível utilizar diretamente os códigos do Unix por razões legais, mas houve um esforço para manter a compatibilidade. Torvalds, ao invés de tentar reinventar a roda, usou o código do antigo Minix para construir o Linux. Raymond descreve essa prática como uma "preguiça construtiva", sendo essa a característica dos "grandes programadores" (1998).

O programador que cria um novo software, torna-se o seu mantenedor, uma espécie de líder e curador, responsável por orientar e engajar a comunidade, e aprovar ou desaprovar mudanças no código-fonte. Abaixo dele estão os desenvolvedores e usuários, que contribuem de acordo com sua capacidade e interesse. Em alguns projetos de maior porte, as comunidades utilizam outras estratégias de governança, como divisão em setores ou cargos eletivos, por exemplo.

O lema “libere cedo, libere frequentemente”, referindo-se ao código-fonte, é a síntese do modelo bazar. Não se trata simplesmente de disponibilizá-lo na internet, mas de torná-lo um método de desenvolvimento, de revisão e de correção de erros. Há uma diferença entre apenas ter o código aberto e ser construído em código aberto. Um software pode ser produzido em um processo opaco, interno a alguma organização e, por alguma razão, ter seu código disponibilizado, mas esse ainda não é o modelo de bazar de desenvolvimento.

Sobre o processo de depuração, ou seja, de revisão e correção de erros, do modelo bazar, Eric Raymond observa:

> Na visão catedral de programação, erros e problemas de desenvolvimento são difíceis, insidiosos, um fenômeno profundo. Leva meses de exame minucioso por poucas pessoas dedicadas para desenvolver confiança de que você se livrou de todos eles. Por conseguinte os longos intervalos de liberação, e o inevitável desapontamento quando as liberações por tanto tempo esperadas não são perfeitas. Por outro lado, na visão bazar você assume que erros são geralmente um fenômeno trivial – ou, pelo menos, eles se tornam triviais muito rapidamente quando expostos para centenas de ávidos co-desenvolvedores triturando cada nova liberação. Consequentemente você libera frequentemente para ter mais correções, e como um benéfico efeito colateral você tem menos a perder se um erro ocasional aparece. (RAYMOND, 1998, tradução nossa)

Linus Torvalds, conscientemente ou não, levou e ainda leva essa prática ao extremo, tratando os usuários do Linux como co-desenvolvedores, incentivando-os a interagir e colaborar. Tem-se, portanto, uma reconfiguração na dicotomia produtor/consumidor, visto que a fronteira entre o usuário e o desenvolvedor se esvai (PEREIRA, 2004). Trata-se também de uma ressignificação do erro, não mais compreendido como uma aberração que deva ser evitada a todo custo, mas algo que inevitavelmente ocorre, que é intrínseco ao desenvolvimento de programas de computador. Para Raymond, o modelo bazar é mais eficiente porque, ao invés de focar no esforço intelectual de poucas mentes, ele aposta na inteligência coletiva de um grande número de usuários e nas diferenças individuais que cada um pode

empregar no processo de depuração. Para sintetizar esse fenômeno, Eric Raymond cunhou a “Lei de Linus”, cujo enunciado, em tradução livre, diz: “dados olhos suficientes, todos os erros são óbvios”.

No modelo tradicional, a ênfase mercadológica se dá ao magnífico momento em que a empresa apresenta o trabalho que desenvolveu ao longo de meses ou anos, e então colhe seus frutos. De certa forma, essa relação assemelha-se ao campo das artes, na qual o artista muitas vezes trabalha isoladamente em seu ateliê ou estúdio e, quando finalizada, apresenta sua obra ao público, para então ser reconhecido. Evidentemente, qualquer tipo de software, livre ou proprietário, recebe atualizações com correções e aperfeiçoamentos, mas no modelo catedral, as mudanças são mais pontuais, enquanto os recursos de maior expressão costumam aguardar o lançamento da próxima grande versão, um novo produto a ser comercializado.

No modelo bazar, sem uma clara fronteira entre produtores e consumidores, a ideia de lançamento do software perde o sentido. Há, consequentemente, uma mudança na noção de progressão e inovação tecnológica, que passa a ser compreendida agora como uma “construção permanente, como um trabalho de *patchwork*, permanentemente inacabado” (PEREIRA, 2004). Um bom mantenedor, portanto, mais do que ter boas ideias ou ser um excelente programador, deve saber fomentar e reconhecer o trabalho de sua base de usuários, além de ter uma visão clara dos rumos do seu projeto, para que possa julgar criteriosamente as implementações da sua comunidade.

Evidentemente, nesse processo, algumas ideias ou códigos serão descartados, seja por terem baixa qualidade, seja por não se adequarem a filosofia do projeto. Quando isso ocorre, o que é bastante comum, o código descartado pode tomar três caminhos distintos, sintetizados por Wilken Sanches (2007): primeiro, ele pode ser abandonado e esquecido; segundo, ele pode ser usado apenas pelo seu autor, ou, terceiro, pode dar origem a um novo software, uma divisão do projeto, um *fork*.

A criação de *forks* é a medida mais custosa e arriscada, pois implica na necessidade de desenvolver uma nova comunidade. Assim, ela só se justifica quando o seu autor acredita que não só há potencial e viabilidade para a sua ideia, como também acredita que ela pode e merece concorrer com o software original. Se por um lado, o surgimento de *forks* proporciona grande variedade de opções para os usuários e representa maior liberdade do desenvolvedor em pôr em prática seus ideais, por outro, eles também dividem comunidades, provocando rivalidades e conflitos, configurando um campo de produção altamente competitivo (SANCHES, 2007).

Há, portanto, uma intensa concorrência nesse meio, tanto dos softwares em si, que concorrem por usuários e colaboradores, como também dos próprios hackers, em um sentido mais pessoal, que buscam reconhecimento por suas habilidades como desenvolvedores, como também é comum entre artistas e arquitetos.

> Uma consequência interessante dos tipos de colaboração desenvolvidos no FLOSS é que os inimigos se encontram trabalhando no mesmo projeto. As empresas que estão em rivalidade pelo menos nominal entre si podem construir seus negócios em torno de código compartilhado ou usar o compartilhamento e desenvolvimento de tal código como forma de desenvolver uma plataforma alternativa ao software proprietário para ganhar participação de mercado. (FULLER; HAQUE, 2008, tradução nossa)

Sério Amadeu da Silveira, sociólogo e defensor do software livre no Brasil, denominou esse fenômeno de "individualismo colaborativo", pois os "hackers do Floss (*Free/Libre Open Source Software*) têm um comportamento extremamente meritocrático. Ao mesmo tempo, seu hiperindividualismo é construído em processos colaborativos" (2010). Talvez essa seja a magia do Linux e do mundo *open-source*: não se trata de manter "todo mundo feliz e cantando ao redor de uma fogueira" (TORVALDS, 2013), mas compreender e explorar a diversidade humana e o potencial agregador da internet e das tecnologias digitais.

2 ARQUITETURA EM CÓDIGO ABERTO

2.1 CRÍTICA AOS ARQUITETOS

Nem toda a argumentação em torno do código aberto na arquitetura parte diretamente de um problema específico. Algumas surgem mais da experimentação, da busca por novas formas de produzir e experienciar o espaço, sempre explorando as possibilidades das tecnologias digitais. No entanto, a maior parte da discussão tem sim origem em um tipo de crítica em comum. Com enfoques distintos, todas elas estão dirigidas a aspectos culturais e estruturais do campo da arquitetura, bem como o papel do próprio arquiteto na produção espacial. Em *OSArc (Open Source Architecture)* (CLAUDEL; RATTI, 2015), norte dessa pesquisa, observa-se uma desaprovação ao que o autor chama de "onipotência do arquiteto". O artigo *A Communism of ideas: Towards an open-source architectural practice* (KASPORI, 2003), por sua vez, argumenta que a profissão do arquiteto está perdendo cada vez mais espaço na construção de habitações e cidades, estando a mercê do mercado. Mesmo que aparentemente contraditórios, tratam-se de pontos de vistas complementares.

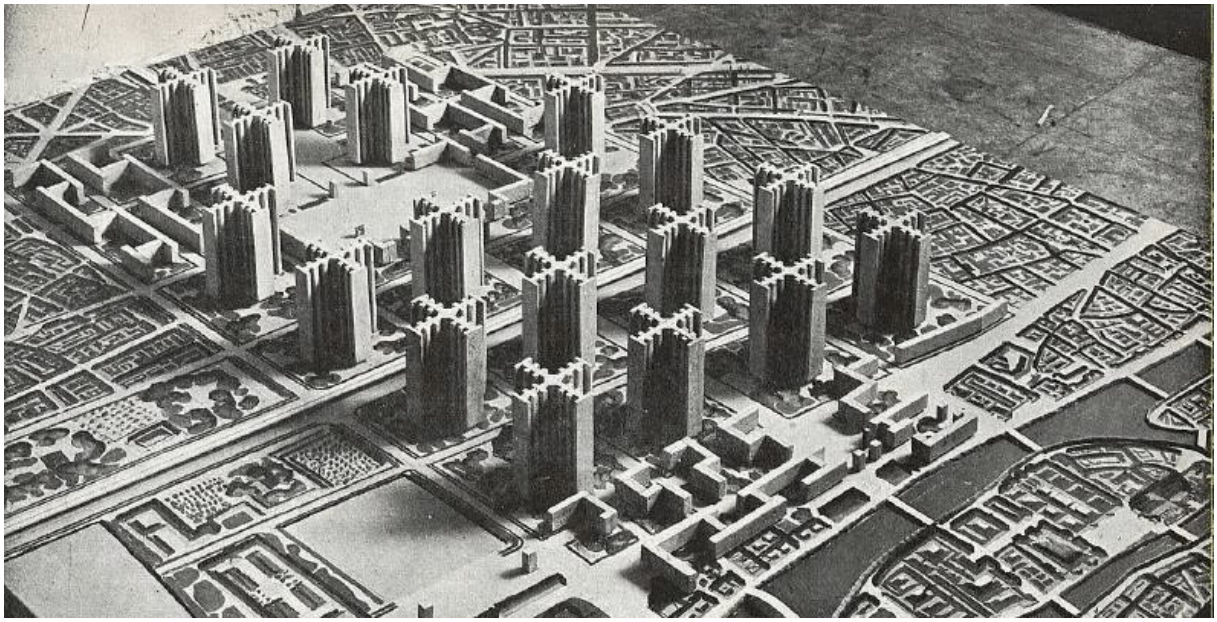
Na informática, o movimento software livre e de código aberto tem como característica a presença de um inimigo comum, um vilão, o grande antagonista. Empresas como Microsoft e Apple, cumprem até hoje esse papel, sendo não só o oposto de tudo o que prega o movimento, como também concorrentes diretos por usuários e recursos. A personificação de um antagonista também se faz presente na produção de Claudel e Ratti. Em *OSArc*, o principal vilão é o arquiteto modernista Le Corbusier.

Nascido em 1887 na Suíça, mas naturalizado francês, Le Corbusier foi arquiteto, urbanista, pintor e escultor, sendo considerado uma das mais influentes personalidades do século XX. Suas convicções foram divulgadas por meio dos seus projetos, executados ou não, mas também por meio dos seus livros,

manifestos, desenhos, palestras, entrevistas, enfim. Uma das propostas de Le Corbusier, A Ville Contemporaine (Figura 3), apresentada em 1922, ainda no início de sua carreira, no Pavillon de l'Esprit Nouveau, mesmo que de caráter utópico, foi escolhida por Claudel e Ratti para exemplificar a crítica ao arquiteto e a grande parte do movimento modernista.

O projeto compreende uma cidade cuidadosamente concebida que seria construída do zero, em uma tábula-rasa, para abrigar 3 milhões de habitantes. Com soluções pautadas na funcionalidade, sob influência de uma ótica industrial, característica do século XX, o projeto da cidade era marcado por arranha-céus cruciformes e uma malha viária rígida e ortogonal. Além disso, "*(...)* cada elemento se encaixava precisamente em um todo uniforme, coerente e eficiente, (...) uma elegante máquina social, cujas engrenagens era a própria arquitetura" (CLAUDEL; RATTI, 2015, pág. 7). A ideia é que esse novo modelo de cidade pudesse ser replicado infinitamente, permitindo uma expansão controlada e planejada da área urbana, ou seja, nada estaria fora do seu projeto.

Figura 2: Maquete da *Ville Contemporaine*.



Fonte: https://www.researchgate.net/figure/Figure-3-3-The-contemporary-city-of-Le-Corbusier-adapted-from-NYU-2009_fig18_319987475

Mas para além das soluções arquitetônicas, o projeto tornou-se controverso porque a cidade projetada seria construída sobre Paris, a qual deveria, portanto, ser demolida. Paris, a tábula-rasa de Le Corbusier, uma cidade construída ao longo de séculos, pela sucessão do trabalho intelectual e manual de diferentes gerações, seria apagada em um único gesto, por um único autor. A justificativa para um planejamento tão centralizado e autoritário era a razão. Os arquitetos modernistas, em tese, detinham o conhecimento e a racionalidade, logo estariam aptos e autorizados a tomar as decisões corretas por um bem maior. Enquanto a arquitetura de natureza vernacular era desprezada, vide a proposta de demolição de Paris, a arquitetura modernista, estabelecida de "cima para baixo", era compreendida como definitiva, infalível, isenta de erros.

Indiscutivelmente um arquiteto-estrela, sua grande influência na mídia e na política, aliada a um momento histórico propício, de reconstrução de nações e governos de bem-estar social, foram fundamentais para um maior empoderamento do profissional da arquitetura. Le Corbusier e o movimento modernista lutaram pela "expansão do arquiteto", pela "onipotência da arquitetura" (CLAUDEL; RATTI, 2015, pág. 7). Os arquitetos se propuseram a reimaginar a sociedade, sendo a arquitetura sua ferramenta de intervenção:

> Ele apresentou o ducesso do design *top-down*: tudo funcionou. Não só funcionou, mas funcionou sem esforço com a graça suave da pura racionalidade – essa foi a autoridade que Le Corbusier assumiu plenamente no dia em que inaugurou seu Pavilhão, sua mão pairando anunciando uma visão pura e indivual... e a sociedade não poderia ajudar, apenas seguir. (CLAUDEL; RATTI, 2015, pág 8, tradução nossa)

A busca por uma "orquestração social" por parte dos arquitetos, importante observar, não surgiu no modernismo, apenas nele se consolidou, alcançou seu auge. Não faltaram planos utópicos de cidades e sociedades na história e na teoria da arquitetura, sempre apoiados em um argumento central coerente com sua época. Com o modernismo os arquitetos ganharam mais espaço no debate

público, construiu-se em torno de Le Corbusier e de sua profissão uma áurea de brilhantismo e de heroísmo.

O primeiro questionamento a esse aspecto da arquitetura modernista diz respeito a sua legitimidade moral. Afinal, pode-se considerar justa uma produção espacial tão centrada em uma único indivíduo? Esse modelo é coerente com os valores de uma sociedade aberta e democrática? Não seria mais correto que os moradores ou usuários pudessem ter uma participação mais ativa nesse processo? Tais perguntas não são feitas diretamente por Claudel e Ratti, mas, por meio da *Ville Contemporaine* e de outros exemplos que antecederam o modernismo, eles induzem o leitor a refletir sobre o tema.

Mas a principal crítica é de natureza prática, funcional. Sem negar uma série de soluções e avanços que a arquitetura moderna trouxe, a exemplo dos Cinco Pontos da Nova Arquitetura, do próprio Le Corbusier, é fato que o desejo utópico por formas puras e um tipo de projeto racional que pudesse servir com eficiência a todos, em todas as escalas, não chegou perto de atingir seus objetivos. A rigidez da arquitetura modernista se mostrou incapaz de compreender a riqueza e a diversidade da vida humana. Independentemente das questões morais, o tempo provou que esses arquitetos não eram detentores da razão e da verdade. Suas obras eram, sim, passíveis de erro.

Tais críticas, no entanto, mesmo que necessárias, foram feitas por muitos outros arquitetos e teóricos pós-modernistas, não sendo, portanto, até aqui, algo inserido apenas no debate sobre arquitetura em código aberto. Além disso, a princípio, não se trata mais de um debate tão em pauta, visto que já existe certo consenso quanto as principais falhas da arquitetura modernista, especialmente do ponto de vista funcional. O tempo permitiu aos arquitetos realizarem uma autocrítica e distanciarem suas obras, principalmente na escala urbana, dessa abordagem tão racionalista e inflexível.

Mas então, estando o modernismo, em tese, no passado, a que problemática se propõe uma arquitetura em código aberto? De fato, a busca por

um planejamento tão centralizado, a medida em que se mostrou falha, foi sendo deixada de lado, e o argumento da racionalidade perdeu força. No entanto, o apelo midiático do campo da arquitetura não só resistiu, como parece ter se consolidado com o passar dos anos. E mesmo que certas soluções não sejam mais aceitas hoje, a ênfase na individualidade do autor, ainda se faz presente, o que Matthew Claudel e Carlo Ratti compreendem como uma extensão do modernismo:

> Ao longo do século 20, o que começou como manifestos dos modernistas se tornou a arquitetura de Venturi (...), a linguística confusa dos desconstrutivistas, as megamonografias de tijolos de Koolhaas e os blogs online de hoje. Em 2009, Bjarke Ingels até lançou um "arquicômico" intitulado Yes Is More – uma *graphic novel* arquitetônica estranhamente atraente na qual o próprio Ingels parece explicar suas ideias e seu trabalho. (CLAUDEL; RATTI, 2015, pág 15, tradução nossa)

Figura 3: Páginas de Yes Is More, *graphic novel* em que Ingels é o autor e o protagonista.

Fonte: https://www.metalocus.es/en/news/bigs-yes-more-now-available-ipad-bjarke-ingels

Para ele, a áurea de brilhantismo e heroísmo ainda persegue o imaginário dos arquitetos, sendo esse um ponto chave de sua crítica, e a que mais se conecta com outros textos sobre código aberto na arquitetura. Enquanto Le Corbusier é o antagonista escolhido por Claudel e Ratti, Kaspori, parte diretamente da arquitetura contemporânea. Na prática, porém, tratam-se duas manifestações de um mesmo fenômeno, o arquiteto-estrela.

Assim como no século XX, essa classe de arquitetos continua buscando se libertar do passado, apoiando-se em uma nova tecnologia ou uma nova filosofia, apresentando à sociedade, e mais especificamente ao seu público, novas formas de construir e de pensar a arquitetura. De certo modo, a ideia de infalibilidade também persiste, porém, o pilar central não é mais a razão e a funcionalidade, mas a inovação e o apelo artístico. Um obra de arte é uma obra de arte e, uma vez assinada, quase sempre é compreendida como pronta e irretocável. Nas palavras de John Habraken, que é um dos coautores de *Open Source Architecture*:

> Este novo profissional queria se libertar do quotidiano, das suas tradições, restrições e limitações. A partir de então, o foco estaria em inovações e uma nova forma de construir. (HABRAKEN, 2006, tradução nossa)

Se por um lado poderia ser interessante a visibilidade midiática do campo, visto que o arquiteto pode influenciar decisões políticas e articular soluções mais efetivas para questões socioespaciais, como fizera o próprio Le Corbusier, e isso precisa ser reconhecido, por outro lado, o campo tem focado tanto na produção de ideias e construções arrojadas e diferenciadas que elas têm sido mais valorizadas que a solução dos problemas em si. Segundo Kaspori (2003) e Claudel e Ratti (2015), os arquitetos têm se isolado em seus discursos,:

> Em uma tentativa apressada de projetar a sociedade e cada um de seus produtos culturais – para finalmente esclarecer o público sobre o que tudo isso significa – o gênio solitário se distanciou do próprio público. O *modus operandi* tem sido cada vez mais projetar edifícios com o máximo de

> visibilidade e importância cultural possível, ao invés de abordar as questões fundamentais relacionadas as habitações humanas, isso sem falar das utopias sociais. (CLAUDEL; RATTI, 2015, pág. 16, tradução nossa)

> As questões espaciais relativas à mobilidade e à segurança, a estagnação do setor de construção de casas e as enormes demandas espaciais para o desenvolvimento vermelho (urbanização) e verde (espaço verde) devem ser resolvidas. Enquanto o país tem grandes e importantes problemas sociais, clamando por intervenção de arquitetos e mediação de arquitetura, a *dutch architecture* continua a se deleitar com a glória do sucesso internacional. (KASPORI, 2003, tradução nossa)

Quando esses autores se valem dos arquitetos-estrelas como manifestações máximas da midiatização e do isolamento do campo, é compreensível que a crítica soe injusta, e talvez até desrespeitosa, com muitos profissionais que são, não só bem intencionados, como também competentes na busca por articulações políticas e financeiras, a fim de promover cidades mais justas, seguras e agradáveis. No entanto, sob uma análise mais atenta de suas obras, em especial a de Kaspori, percebe-se que não se trata de um questionamento de cunho pessoal, mas uma crítica a características culturais e estruturais do campo da arquitetura em geral.

Não importa se os projetos são museus de volumetrias futuristas, belíssimas salas de estar decoradas, ou ainda protótipos de habitações modulares flutuantes. É natural que os arquitetos, famosos ou menos conhecidos, queiram ser inovadores, fora da caixa, se destacando nas redes sociais e sendo prestigiados por suas qualidades intelectuais e artísticas. Isso diz respeito não só à sua satisfação pessoal, o que é válido, mas também à sua prosperidade financeira. O modelo autoral e individual de produção, por propiciar uma forma de ascensão profissional eficiente e, idealmente, meritocrática, tem favorecido um certo conformismo da classe. Assim, mesmo que existam muitas iniciativas que olhem com seriedade para as mazelas sociais, elas ainda são pontuais.

Assim, em síntese, tal isolamento ocorre por dois aspectos que se complementam e se reforçam: o fator cultural, relacionado ao imaginário do arquiteto, sua autopercepção enquanto profissional; e o fator estrutural, relacionado aos mecanismos de incentivo. Ao invés de buscarem novas formas de serem ouvidos e atuarem mais efetivamente na solução de questões que tradicionalmente competem ao campo da arquitetura, os arquitetos mantém-se imersos em um ciclo de inovação pela inovação, como mera ferramenta de autopromoção, uma eterna "reinvenção da roda" (KASPORI, 2003). Esse fenômeno é evidente na grande quantidade de escritórios que se submetem a concursos, por exemplo, mesmo que pouquíssimos desses tenham possibilidade de serem executados.

Na prática, como um todo, o campo da arquitetura perdeu o espaço e a autoridade que tivera no modernismo, atuando, hoje, majoritariamente sob a demanda dos grupos que os financiam. Sobre essa relação com o mercado, é interessante notar que, apesar do grande volume de novas ideias no campo da arquitetura, no mundo real, a inovação costuma ser evitada pelos investidores. Tem-se como resultado uma classe de profissionais muito criativa, gerando novas ideias incessantemente, mas uma produção conservadora, pautada em frias análises de risco (KASPORI, 2003). Para Claudel e Ratti:

> através de livros, filmes, innternet e muita força de vontade, a ideia cultural e a autoconcepção do arquiteto tiveram um grande sucesso, enquanto a própria arquitetura fracassou – tanto como modelo de negócios quanto como ferramenta para mudanças sociais benéficas. (CLAUDEL; RATTI, 2015, pág. 16, tradução nossa)

Em coerência com essa crítica, a arquiteta Denise Nascimento, à luz da teoria social de Pierre Bourdieu, diz: “o arquiteto (...) coloca-se afastado do funcionamento da realidade contemporânea capitalista e política, permanecendo na esfera da retórica e não da ação” (2017, pág. 289). Ela distingue duas classes de arquitetos, os mais renomados, detentores de maior capital intelectual, ou seja,

status, reconhecimento, e os mais desconhecidos, que focam sua produção na obtenção de capital econômico, ou seja, a busca pelo sucesso financeiro. Para ela, essa busca de capitais abre espaço para que outros interesses interfiram e se sobressaiam na produção arquitetônica:

> Aliado à passividade dos arquitetos de se relacionar com o real, na medida em que se resguardam apenas pelo capital econômico (profissional) e intelectual (simbólico), o habitus coletivo deixa uma passagem para que outros campos (econômico, político, imobiliário) exerçam seus poderes por meio da imposição de suas regras e práticas construídas em cada um deles. (NASCIMENTO, 2017, pág. 289)

> A cidade expressa diferenciadamente a produção daqueles arquitetos que carregam maior capital econômico, evidenciando as suas amarras e sujeições (a produção genérica e repetitiva de torres residenciais e comerciais incorporadas e comercializadas por grandes construtoras). Ao contrário, os arquitetos com maior capital intelectual explicitam a linguagem de ruptura, mas os desenhos passam a ser mais importantes do que as construções dos edifícios. (NASCIMENTO, 2017, pág. 287)

Figura 4: produção genérica em habitações do Minha Casa Minha Vida.

Fonte: https://www.cartacapital.com.br/politica/minha-casa-minha-vida-governo-quer-trocar-financiamento-por-aluguel/

Para que a arquitetura possa reconquistar o seu espaço e ter um impacto positivo na sociedade, talvez seja necessário destruir o mito do arquiteto enquanto visionário (KASPORI, 2015). Diante dos dois fatores, cultural e estrutural, não seria suficiente apenas uma mudança de mentalidade por parte dos arquitetos, mas sim uma completa reestruturação do campo, um modelo de produção coletivo, alternativo à centralidade na individualidade do arquiteto.

Nesse sentido, não é estranho que Claudel, Ratti e outros arquitetos abordados na presente pesquisa, tenham buscado referências na informática e, especialmente, no movimento software livre e de código aberto. Afinal, além da cultura do movimento, há também uma metodologia de desenvolvimento e uma estrutura de incentivos que possibilita o trabalho coletivo, eficiente e, segundo Sérgio Amadeu da Silveira (2004), meritocrático.

2.2 DO VERNACULAR AO BAZAR

Entendendo a dinâmica do movimento software livre e de código aberto, suas motivações, metodologias e razões de seu sucesso, e diante das críticas levantadas relacionadas à ênfase nos discursos pomposos e na individualidade do autor, bem como à consequente perda de relevância dos arquitetos diante de problemas sociais e de autoridade diante do mercado e dos investidores, é compreensível a esperança por respostas no campo da informática. Como já observado, mais do que a habilidade e a experiência de um ou outro desenvolvedor, a força de projetos como o Linux, por exemplo, está em seu modelo de desenvolvimento. Mas afinal, é possível uma produção arquitetônica que tenha como pilar comunidades semelhantes às que desenvolvem os softwares livres? De que forma elas se materializariam? Qual seria o papel dos arquitetos nesse contexto?

Primeiramente, devido a natureza aberta e social dos softwares livres, a busca por um paralelo entre o modelo bazar e a arquitetura exige uma

compreensão mais ampla da produção arquitetônica, considerando não só o trabalho dos profissionais do campo, mas também o processo de autoconstrução. Entende-se por autoconstrução a:

> provisão de moradia e/ou de espaço público onde moradores de posse de uma área urbana, obtida no mercado formal ou informal, decidem, constroem ou interferem por conta própria no espaço, utilizando seus próprios recursos e, em vários casos, mão de obra familiar, de amigos ou ainda contratada. (NASCIMENTO, 2017, p.291)

Apesar de marginalizada, a autoconstrução representa parte significativa da construção civil no mundo, se destacando especialmente em países menos desenvolvidos. No Brasil, de acordo com um estudo produzido pela Booz Allen Hamilton, aproximadamente 77% das unidades habitacionais produzidas, reformadas ou ampliadas se dão nesse regime.

Conceber e produzir espaços independe da atuação do arquiteto, é uma "manifestação cultural da necessidade humana, não apenas de abrigo, mas também de status, identidade e prazer" (CLAUDEL; RATTI, 2015, pág. 22). De acordo com o arquiteto e teórico holandês N. John Habraken (1998), o ato de construir e de habitar faz parte de um único processo, são uma coisa só, uma relação natural e literalmente construtiva do espaço edificado. Como o próprio nome induz, a autoconstrução é um processo autônomo e espontâneo, próprio da condição da nossa espécie.

A arquitetura vernacular, a autoconstrução, não ocorre paralelamente ao campo da arquitetura, ela é a sua essência. A história da arquitetura, e não a história dos arquitetos, é coletiva e anônima, fruto da interação e da socialização humana. Ela não é marcada pela individualidade do arquiteto-artista, mas por uma profusão de saberes transmitidos e aperfeiçoados por meio de uma sucessão de pequenas decisões projetuais que resultam em uma produção mais adaptada ao contexto espacial, social e cultural na qual se insere.

> (...) uma pessoa pode projetar sua casa para ser parecida com a dos seus vizinhos, mas com pequenas modificações e melhorias. Depois que um projeto é construído, ele é avaliado pela comunidade, mesmo que inconscientemente, e os projetos seguintes vão se modificando e inovando. Assim, a arquitetura se propaga e evolui com base em tipologias, informações compartilhadas e experimentações sutis (...). (RATTI, 2015, pág. 111, tradução nossa)

É inevitável observar o contraste com o modelo de produção dos arquitetos, criticado por Claudel, Ratti e Kaspori. Do ponto de vista dos arquitetos mais renomados, enquanto sua produção busca sempre um estado de perfeição, de isenção de erros, artisticamente irretocável, a produção vernacular sempre ocorreu em um processo incessante de tentativa e erro. E enquanto as "habitações sociais", por exemplo, são definidas autocraticamente pelos construtores formais, resultando em edifícios cada vez mais genéricos, alheios às necessidades e interesses dos moradores, impondo uma relação passiva com o espaço edificado e com o próprio ato de morar, a arquitetura vernacular emerge de baixo para cima, tendo suas soluções concebidas diretamente pelos próprios moradores. Sobre isso, Nicholas Negroponte aponta:

> a arquitetura, particularmente a habitação, tem sido inadequada e não responde às necessidades e desejos de seus usuários... o projeto da habitação está nas mãos erradas, isto é, nas mãos de um "profissional" externo e não do residente. (NEGROPONTE, 2007, tradução nossa)

Há na autoconstrução uma evidente semelhança com o modelo bazar definido por Raymond, que assume o erro, o envolvimento e a experimentação coletiva como pilar fundamental do seu processo inovativo, justificando o lema "havendo olhos suficientes, todos os erros são óbvios". Outra ponto em comum é que tanto na arquitetura quanto na informática, há uma compreensão diferente da relação entre produtor e consumidor, pois do mesmo modo em que nos softwares livres há a possibilidade e o incentivo para que usuários com algum tipo de conhecimento se envolvam ativamente com o desenvolvimento das

ferramentas que utilizam, na produção de espaços urbanos e principalmente de moradias, o próprio usuário se torna o arquiteto, reformando sua casa, ampliando ou criando novos ambientes, pintando, decorando, abrindo ou fechando janelas, ou ainda organizando mutirões de restauração e requalificação de vias, praças, enfim.

Mais que uma ou outra casa, as cidades são o produto por excelência da autoconstrução. Referindo-se às cidades antigas da Itália, mas suas constatações compreendem praticamente qualquer cidade, especialmente as que se originaram sem um planejamento centralizado, Claudel e Ratti observam:

> (...) elas compartilham uma qualidade comum: detalhes intrincados e variedade infinita. Cada cidade é um agregado de camadas, de histórias, de vozes, de linhas familiares e lutas pelo poder. É uma arqueologia da experiência, cimentada na arte, nos edifícios e nas praças. (CLAUDEL; RATTI, 2015, pág. 20, tradução nossa)

No campo do urbanismo, a espontaneidade da autoconstrução resulta em uma riqueza e diversidade espacial que nunca poderia ser concebida por uma única mente humana. Trata-se de um artefato necessariamente coletivo, que reúne qualidades, experiências e visões de mundo, que não apenas se somam, mas interagem entre si, tanto no campo das ideias, do conhecimento compartilhado, como no campo material, por meio das relações espaciais entre edifícios na malha urbana, conferindo identidade ao lugar. Como já dito, a história da arquitetura, e também das cidades, ao longo de milhares de anos, é coletiva, fruto da interação e da socialização humana.

Na escala arquitetônica, as catedrais góticas são bons exemplos da riqueza do trabalho coletivo, da “infinita variedade e detalhes intrincados”. Nelas, várias pessoas ao longo de décadas puderam acrescentar um pouco de sua identidade, de seus valores estéticos e habilidades artísticas, mas nunca faltou coesão, seja pela própria tipologia, seja por um “espírito da época”. Fato é que, mesmo sendo uma obra construída coletivamente, na qual várias pessoas tinham autonomia

para intervir, havia um ímpeto comum, possivelmente associado à figura de um líder ou um arquiteto, que ao invés de assumir para si cada detalhe dos vitrais, por exemplo, estabelecia as principais diretrizes da obra. Por ironia, a construção das catedrais góticas se deu de maneira mais próxima ao modelo bazar do que ao modelo catedral de desenvolvimento.

O reconhecimento e a valorização dessa natureza coletiva e autônoma da arquitetura, seja ela na escala urbana, seja na do próprio edifício, implica em um ponto fundamental: a princípio, a arquitetura e os softwares em código aberto têm muito mais semelhanças do que se poderia assumir em um primeiro momento. Buscar inspirações e referências nesse modelo de desenvolvimento, portanto, não implicaria em um rompimento custoso, mas um desnudamento do próprio campo. Mas se assim for, qual seria o papel e a importância do arquiteto?

Primeiramente, apesar da necessidade de reconhecimento de certas qualidades da autoconstrução, é preciso considerar também seus efeitos negativos. O desenvolvimento baseado na tentativa e erro tem seu valor histórico e cultural, sendo um eficiente motor de inovação e uma possibilidade de empoderamento diante da falta de acesso a um profissional. No entanto, esses erros não podem ser romantizados. Não são poucos os casos de equívocos projetuais que dificilmente aconteceriam na presença de arquitetos, como a ausência de janelas em dormitórios, por exemplo, que possibilitam o desencadeamento de uma série de doenças.

Após o auge do modernismo, a questão da participação do usuário veio a tona no discurso arquitetônico. Muitas ideias e experimentações surgiram, como as de Nicholas Negroponte, Cedric Price, Lucien Kroll, Christopher Alexander, John Habraken, entre muitos outros. Claudel e Ratti porém enfatizam que há uma diferença considerável entre estratégias de participação coletiva e processos verdadeiramente autônomos. Os métodos de participação e inclusão do usuário muitas vezes buscam o consenso, mas esse objetivo não só é difícil de ser alcançado, como muitas vezes, na prática, não passam de estratégias de

manipulação, conferindo um ar democrático para decisões autocráticas, sendo muito usadas por governos e instituições para validar decisões tomada em altos escalões.

> "Design em comitês" não é uma solução adequada: tal abordagem é sempre limitada pela aprovação do menor denominador comum – o fato de que todos devem concordar em todas as partes de um processo de design, ou, mais realisticamente, em como esses processos legitimadores são fiados e manipulados por diversos interesses. Mais importante é concentrar-se em ampliar as esferas de responsabilidade das pessoas e, portanto, a motivação, o compromisso e a agência em relação ao design e à habitação do ambiente urbano. (FULLER; HAQUE, 2008, tradução nossa)

Os processos autônomos emergem justamente da soma das "diferentes agendas e aproximações". As características formais da autoconstrução, especialmente na escala urbana, configuram um modelo de contínuas sobreposições de intervenções individuais, como um *patchwork* (PEREIRA, 2004). Desse modo, uma arquitetura em código aberto não deve olhar para metodologias de produção pautadas na harmonia entre as partes, com "todo mundo feliz e cantando ao redor da fogueira", como brincou Linus Torvalds, mas sim na potencialização de processos autônomos, concebidos "de baixo para cima".

Sobre o papel do arquiteto, foram percebidas duas abordagens principais, porém não antagônicas. A primeira delas, enfatizada por Claudel e Ratti, compreende o arquiteto como um agente externo à autoconstrução, possibilitando uma coordenação dessa dinâmica sob um ponto de vista privilegiado, de modo semelhante ao que acontecia nas catedrais góticas. O foco do profissional estaria, portanto, na criação de condições para *que a concepção e a produção pudessem acontecer de forma segura e otimizada. Na analogia com o software livre, os arquitetos estariam para os mantenedores, atuando majoritariamente no engajamento e na coordenação da comunidade, mas* ainda mantendo o direcionamento do projeto. Em suas palavras:

> O produto do arquiteto, então, não seria necessariamente edifícios ou documentos de construção, mas começar, terminar e coordenar o processo pelo qual o código-fonte arquitetônico é compartilhado, adaptado e executado. (CLAUDEL; RATTI, 2015, pág. 115, tradução nossa)

> Seu objetivo é transformar a arquitetura de um mecanismo de entrega imutável de cima para baixo em um sistema ecológico transparente, inclusivo e de baixo para cima – mesmo que ainda inclua mecanismos de cima para baixo. (CLAUDEL; RATTI, 2015, pág. 132, tradução nossa)

Apesar da analogia simples, ainda não há clareza sobre como essa atuação funcionaria na prática. Afinal, o que seria o código-fonte da arquitetura? É possível viabilizar um novo escopo profissional de modo a transformar estruturalmente o campo, transcendendo iniciativas pontuais? Há uma estrutura de incentivos possível para essa proposta?

A outra abordagem possível coloca os arquitetos no mesmo patamar dos autoconstrutores, inserindo-os diretamente nessa rede colaborativa. Enfatizando "um processo de conscientização crescente", ou seja, uma mudança cultural no campo, Kaspori (2003) defende que é preciso repensar as formas de inovação, deixar um pouco de lado a questão da autoria individual e considerar a arquitetura como um processo evolutivo, interdisciplinar e inclusivo, integrando profissionais e autoconstrutores, compreendendo "a arquitetura não apenas como um objeto estético ou peça de exibição, mas também como um processo de aprendizagem e um assunto para discussão" (2003). Para ele:

> (...) o código aberto fornece um modelo de organização para o desenvolvimento coletivo de soluções para questões espaciais que envolvam habitação, mobilidade, espaços verdes, renovação urbana e assim por diante. São questões complexas que pressupõem uma abordagem interdisciplinar; na verdade, elas só podem ser resolvidas com cooperação. O código aberto pressupõe que essas ideias sejam divulgadas e disponibilizadas para outros, que por sua vez podem melhorá-las. Dessa forma, o design muda de uma ação pontual para uma espécie de processo evolutivo. (KASPORI, 2003, tradução nossa)

Como visto anteriormente, o modelo bazar promove um ambiente colaborativo, mas tem como princípio motriz interesses individuais. A partir do momento em que soluções específicas podem ser utilizadas noutros contextos, e é o que costuma acontecer com softwares e outros tipos de informações e conhecimentos, há uma aproximação espontânea de novos colaboradores, nascendo então uma comunidade. Em outras palavras, para que haja colaboração, é preciso que exista ou se promova um interesse em comum. Para Kaspori (2003, "a identificação dessa base de usuários é um passo importante no desenvolvimento prático de uma arquitetura em código aberto". Em suma, trata-se de uma das principais demandas do mantenedor de um software, como lembra Raymond:

> Quando você começa a construir uma comunidade, o que você precisa apresentar é uma promessa plausível. Seu programa não precisa funcionar particularmente bem. Pode ser grosseiro, cheio de bugs, incompleto e mal documentado. O que não pode deixar de fazer é ser executável e convencer potenciais co-desenvolvedores de que o projeto pode evoluir para algo realmente interessante em um futuro próximo. (RAYMOND, 1998, tradução nossa)

Assim, apesar de abordagens com enfoques distintos, há também uma certa convergência. Ao invés de uma ou outra possibilidade de atuação profissional, sugere-se que o arquiteto busque se inserir em comunidades colaborativas, pelo simples fato dessas terem um grande potencial de inovação e solução de problemas mais complexos. Mas também sugere-se que busque compreender o papel e a dinâmica dessas redes, de modo a melhor manipulá-la e extrair o seu potencial. Em outras palavras, na mesma medida em que o arquiteto deve atuar de maneira mais colaborativa e em rede, buscando a construção coletiva e distribuída de conhecimento, ele deve também ter senso crítico para promover a criação e o desenvolvimento dessas comunidades, não sendo apenas parte da espontaneidade, mas catalizando esse processo, estabelecendo

prioridades, tomando decisões, definindo rumos, enfim. O próprio Linus Torvalds, mantenedor da maior comunidade de software livre do mundo, nunca deixou de ser um programador ativo, contribuindo com outros projetos para fins majoritariamente pessoais, como por exemplo um aplicativo de planejamento e registro de mergulho, além do próprio Linux.

Para além da proposta de transformação cultural do campo, não se pode desconsiderar a questão da viabilidade econômica, relacionada à estrutura de incentivos. O conhecimento, bem como seu compartilhamento, é o pilar de uma nova prática arquitetônica (KASPORI, 2003). Mesmo que pareça óbvio, porém, é preciso fazer uma distinção: enquanto o conhecimento arquitetônico refere-se a um conjunto de informações organizadas com potencial de aplicação em diversas condições, a ideia de projeto refere-se à aplicação desse conhecimento em situações específicas. Pode-se pensar, por exemplo, uma tipologia como uma forma de conhecimento arquitetônico, assim como a sua construção em um terreno em um determinado local é evidentemente um projeto, uma aplicação prática desse conhecimento. Quando um autoconstrutor observa seus vizinhos em busca de inspirações, ele está se utilizando desse conhecimento, porém adaptando-o à sua necessidade.

Esse processo de transformação de um conhecimento genérico em uma solução específica, além de ocorrer pelos próprios autoconstrutores, é também a base do modelo de negócios dos arquitetos: a prestação de serviços. Esse modelo, por sinal, é também o que possibilita o lucro de grandes empresas que trabalham com softwares em código aberto. Sérgio Amadeu Silveira diz:

> O modelo de negócios do software livre é baseado em serviços. O modelo de software proprietário é centrado em licenças de propriedade. O primeiro, por expor seu código-fonte, busca vender desenvolvimento, capacitação e suporte especializado. O segundo vive do aprisionamento dos seus clientes ao pagamento de licenças de uso. O primeiro modelo exige que a empresa inove permanentemente para manter sua clientela. Já o segundo utiliza a vantagem das enormes dificuldades de mudança de sua solução fechada. (SILVEIRA, 2004, pág. 65)

Assim, a princípio, a maior mudança da arquitetura em código aberto não está relacionada ao projeto em si, no sentido de adaptação a situações muito específicas, pois esse modelo de negócios é majoritário tanto no campo da arquitetura quanto nos softwares livres. Busca-se, na verdade, o desenvolvimento colaborativo de soluções com potencial de uso também coletivo, pois é aí que se torna possível o interesse de outros usuários e a consequente configuração de uma comunidade de colaboradores entorno dessa.

A matéria e a informação têm características distintas, impondo limites a certas analogias. No entanto, a medida em que as tecnologias digitais se tornam cada vez mais presentes no dia a dia de arquitetos e autoconstrutores, a ideia de uma arquitetura em código aberto tende a se tornar mais palpável. Muito se falou sobre a construção de uma rede colaborativa. No entanto, não há possibilidade prática de se fomentar tal rede, uma grande comunidade de compartilhamento de conhecimento e soluções arquitetônicas, apenas em uma vizinhança. Ou melhor, até poderia, mas seria uma compreensão limitada da ideia de abertura.

As comunidades de softwares livres cresceram e se mostraram eficientes não só pela evolução dos computadores, mas principalmente pelo interconexão mundial de diferentes pessoas e culturas proporcionada pela internet. Se a arquitetura é fruto da interação e da socialização humana, assim como as comunidades dos softwares livres, seria inconcebível, portanto, uma arquitetura em código aberto fora dessa rede global, alheia a era digital.

2.3 O DIGITAL NA ARQUITETURA

A definição mais conhecida para uma arquitetura em código aberto é a que foi publicada na revista Domus em 2011 e hoje está disponível na Wikipedia. Não se trata de uma definição tão sucinta: ela descreve o conceito mais pelo somatório de algumas características do que por uma singularidade:

> A arquitetura em código aberto é um paradigma emergente que defende novas formas de imaginar e desenvolver espaços virtuais e reais em uma infraestrutura universal. Partindo de referências tão diversas, como cultura de código aberto, design modular, arquitetura de vanguarda, ficção científica, teoria da linguagem e neurocirurgia, ela busca uma abordagem inclusiva do design espacial, uma apropriação colaborativa do processo e das ferramentas de design, por profissionais e cidadãos comuns. (OPEN-SOURCE ARCHITECTURE, 2021, tradução nossa)

No entanto, pode-se inferir por meio de produções teóricas e também de algumas iniciativas práticas que em síntese a arquitetura em código aberto é uma tentativa de implementação do modelo bazar em todo o ciclo de desenvolvimento do espaço. Trata-se da expansão e da catalização de processos colaborativos e autônomos, intrínsecos à arquitetura, por meio das tecnologias de informação e de comunicação (TIC's). Esse modelo só terá sua completude quando alçado à escala global. O bazar é necessariamente aberto, não podendo estar localmente limitado, como ocorre na produção vernacular. A internet, portanto, torna-se o principal elemento condutor dessa proposta.

Em *Urban Versioning System 1.0*, manifesto em defesa de uma arquitetura em código aberto, publicado em 2008, Usman Haque e Matthew Fuller afirmam:

> (...) construir edifícios é um esforço substancialmente colaborativo, sempre envolvendo equipes e vários tipos de expertise e tomada de decisão. Tudo o que é necessário para libertar a construção é tornar seu repertório de colaboração mais expansivo. (FULLER; HAQUE, 2008, tradução nossa)

A medida em que modelos virtuais tornam-se cada vez mais precisos, cada vez mais próximos do mundo físico, quase como se já existissem, o compartilhamento da arquitetura pode se tornar mais efetivo (CLAUDEL; RATTI, 2015). Enquanto em processos colaborativos tradicionais, relacionados à autoconstrução, os aprimoramentos ocorrem muito lentamente no próprio espaço edificado, seja por meio de reformas, seja no aprendizado para execução em novas construções, no meio virtual, onde a arquitetetura ainda reside no campo da

informação, esses aprimoramentos ocorrem mais rapidamente e com menor custo. Ao fim, tanto um projeto não executado quanto um código-fonte são informações, logo devem responder às mesmas leis da natureza.

> O compartilhamento do conhecimento maximiza o desenvolvimento de bens por aproximar-se o máximo possível da exploração das potencialidades da rede e das características inerentes aos bens informacionais. (SILVEIRA, 2010).

Como visto anteriormente, há uma diferença entre o conhecimento e o projeto em si, ou em outras palavras, entre soluções genéricas, amplas, e soluções específicas, contextuais. Não se tratam de coisas totalmente distintas, visto que muitos projetos concebidos para um tipo de terreno, por exemplo, podem ser estudados e implementados noutro. Mas ter essa percepção dialética é importante para compreender o potencial de alguns empreendimentos.

A respeito uma fachada de brises, apenas para fins ilustrativos, a disposição desses pode ser desenhada por uma equipe de arquitetos, tendo como parâmetros a carta solar, o sentido predominante dos ventos, valores estéticos e também as exigências do programa solicitado. O tamanho, a localização e a orientação dos brises é, evidentemente, uma solução específica ao seu contexto, um projeto de arquitetura em seu sentido convencional. Quando essa mesma fachada, ao invés de desenhada diretamente por uma equipe, é definida por um script que recebe os parâmetros citados e automaticamente devolve um desenho, esse script torna-se o artefato mais valioso desse processo, enquanto o desenho atua apenas como meio de comunicação direcionado aos construtores.

O premiado arquiteto brasileiro Guto Requena, no livro *Habitar híbrido: subjetividades e arquitetura do lar na era digital* (2019, pág. 52), observa uma relação de simbióse entre o homem e a máquina, na qual o homem é o "responsável por desenvolver o processo e escolher o resultado final", e a máquina é "responsável pelas diversas variações formais no processo de

modelagem paramétrica". Reforçando o papel do projetista, a prof. dra. Cristina Cardoso (2005) acrescenta que:

> Sem o aporte de informações dadas inicialmente pelo projetista, o computador não inicia sozinho o processo de projetação, aleatoriamente. É preciso que o projetista defina quais as informações são importantes e necessárias para o desenvolvimento do processo, e essas são então tratadas pelo computador. (CARDOSO, 2005, pág. 136)

Percebe-se, portanto, que a ideia de projeto se amplia, transcende a estaticidade do desenho e da sua reprodução fidedigna, compreendendo também soluções que, por meio das tecnologias digitais, são cada vez mais versáteis e capazes. Por se tratar de uma informação, um código, que pode ter serventia em diferentes contextos, tem-se um cenário propício para configuração de uma comunidade em código aberto.

> Códigos e scripts de código aberto permitem que comunidades de design compartilhem e comparem informações e otimizem coletivamente a produção por meio de componentes modulares, acelerando o acúmulo de conhecimento compartilhado. BIM (Building Information Modeling) e outras ferramentas e metodologias de colaboração permitem a alocação interdisciplinar de informações de projeto e a integração de uma variedade de plataformas e escalas de tempo. (OPEN-SOURCE ARCHITECTURE, 2022, tradução nossa)

A inserção das tecnologias digitais na arquitetura fica ainda mais evidente quando se considera os efeitos da Quarta Revolução Industrial, em especial as técnicas construtivas baseadas nas ferramentas de fabricação digital, como as impressoras 3D e as fresadoras laser CNC. Retomando o exemplo da fachada de brises, as informações geradas pelo script possibilitam que cada unidade apresente um formato único e sejam produzidas uma única vez. Há nisso uma maior liberdade criativa, visto que as amarras da produção em série são superadas.

Um passo importante para a democratização da fabricação digital, tornando-a menos dependente de altos investimentos, se deu com o surgimento dos Fab Labs ao redor do mundo. Concebidos originalmente no MIT, os Fab Labs, segundo definição da Fab Charter, consistem em "uma rede global de laboratórios locais, que possibilitam a invenção e fornecem acesso à ferramentas de fabricação digital", dispondo de "apoio operacional, educativo, técnico, financeiro e logístico" e "acesso aberto (e agendado) para pessoas e também projetos". Com isso, na visão de Claudel e Ratti, os Fab Labs permitiram que pessoas comuns “modificassem ou ‘hackeassem’ o mundo ao seu redor, em vez de absorverem passivamente informações ou produtos” (2015, pág. 85).

Figura 5: Laboratório na FAU-USP, onde fica o Fab Lab SP, o primeiro do Brasil.

Fonte: https://fotografia.folha.uol.com.br/galerias/1605480630054180-fab-labs-em-sao-paulo

Há um ponto interessante que, mesmo que não pareça ter relação direta com a ideia de código aberto, é coerente com os princípios que originaram o movimento: o financiamento coletivo. Trata-se de uma extensão natural de valores da cultura hacker. Na informática, a compilação de um código, ou seja, a sua transformação em um programa utilizável não implica em custos extras, pois ambos se mantém no campo da informação. Na arquitetura, por outro lado, só há execução quando se tem investimento. O financiamento coletivo, nesse contexto,

surge como uma adaptação necessária, uma possibilidade de maior liberdade, em oposição à apatia e à dependência do campo da arquitetura diante do mercado, como observou Kaspori em 2003.

O financiamento coletivo, ainda, pode surgir para viabilizar a própria concepção do projeto, possibilitando novas formas de interação de cidadãos com o espaço urbano.

> O financiamento de projetos privados passa cada vez mais para o domínio público, oferecendo propriedade em massa em vez de individual, enquanto o financiamento de projetos públicos pode ser derivado de estruturas mais flexíveis e responsivas do que simples taxas ou impostos. A arquitetura de código aberto deve ter um apelo especial para construtores totalmente fora da economia convencional, como posseiros, refugiados e militares. (OPEN-SOURCE ARCHITECTURE, 2022, tradução nossa)

Em coerência com as características da arquitetura como um todo, compreendendo-a enquanto processo de aprimoramento contínuo, Fuller e Haque (2008) levantam um questionamento importante sobre o conceito aqui discutido:

> Embora alguns tenham argumentado que os desenhos arquitetônicos podem ser considerados “código-fonte” e, portanto, é o processo de design que deve ser aberto, um dos aspectos mais interessantes do software de código aberto é a intercalação contínua de processos de produção, implementação, uso e reaproveitamento, tudo isso pode deve ser aberto – não apenas um “design aberto” que é implementado de maneira fechada. (HAQUE, FULLER, 2008, tradução nossa)

A materialização da arquitetura no espaço físico, portanto, não deve ser um ponto final, a morte da concepção. Se os softwares são corrigidos e adaptados durante toda a sua existência, uma arquitetura em código aberto, mesmo com as limitações da matéria, deve buscar essa editabilidade também no espaço físico. Para os autores, remetendo à visão de John Habraken, é importante pensar em estratégias de abertura para a arquitetura em si, afinal:

> (...) não faz sentido ter um processo de projeto “aberto” que resulte em uma entidade estruturalmente “fechada”. A arquitetura que é produzida através de um processo autenticamente aberto nunca termina: não há distinção entre design e habitação. (HAQUE, FULLER, 2008, tradução nossa)

Primeiramente, é importante atentar que arquiteturas contemporâneas não são necessariamente estáticas no tempo. De fato a maioria das casas e espaços físicos apresentam essa características, mas cada vez mais se popularizam soluções que se moldam em função de condições variáveis. A interatividade do espaço não surgiu com as tecnologias digitais, mas sem dúvidas foi por elas potencializada. Dados coletados em sensores de temperatura, iluminação, ventilação, presença, toque, som, qualidade do ar, umidade, entre muitos outros, podem servir como parâmetros de entrada para algoritmos que controlam diretamente o espaço físico. A arquitetura vira um “processo contínuo e evolutivo, em oposição à metodologia ‘executar e esquecer’ da arquitetura tradicional” (OPEN-SOURCE ARCHITECTURE, 2022). Na fachada em brises, por exemplo, porque não criar uma estrutura móvel, controlada por software, que tenha como parâmetros não apenas as informações do lugar, mas também a estação do ano, a hora do dia, ou o uso em um dado momento?

O arquiteto indiano Mahesh Senagala, da Universidade do Texas em San Antonio, para definir essa tendência, cunhou o termo *time-like architecture*. Para ele, além das três dimensões espaciais, o tempo deveria ser incorporado à arquitetura, de modo que ela não seria apenas uma forma, mas também um comportamento, tal qual um organismo vivo. No Brasil, o Estúdio Guto Requena é uma referência na exploração da interatividade no espaço edificado. Sobre o Bar Emotividade, um dos projetos do escritório, construído em 2015 a pedido da Casa Cor, Requena explica em seu site:

> Os visitantes são convidados a beber uma cerveja e contemplar uma obra interativa com comportamento visual e sonoro gerados em tempo real pelas emoções das pessoas na internet. A obra "Cartografia Emotiva"

> transforma o bar num espaço híbrido, um imenso observatório de emoções. Uma projeção mapeada (de 30m X 10m) apresenta graficamente a coleta em tempo real feita na internet (Twitter e Instagram) de hashtags de emoções. Uma música generativa (produzida pelo computador) é criada também em tempo real como resposta a essas hashtags. (BAR EMOTIVIDADE, 2022)

Figura 6: Painel interativo no Bar Emotividade.

Fonte: https://gutorequena.com/baremotividade/

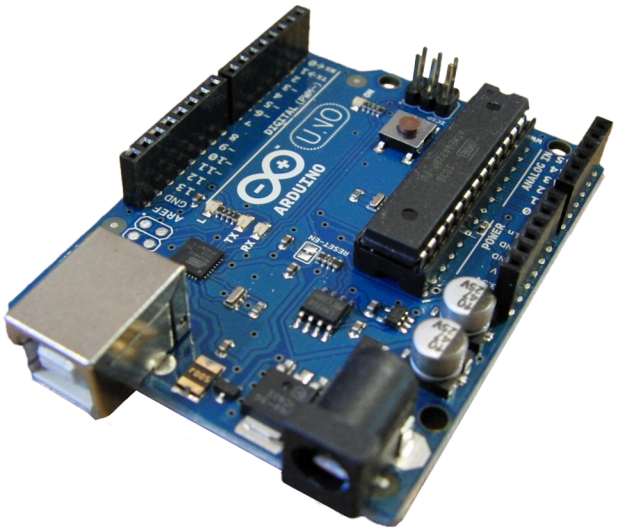
Assim como os Fab Labs promoveram a democratização do acesso ao conhecimento e aos equipamentos necessários para a confecção de artefatos em fabricação digital, o Arduino é apresentado por Claudel e Ratti como uma possibilidade de democratização da arquitetura interativa. Arduino é uma plataforma aberta de prototipação eletrônica que, na prática, consiste em uma placa com um microcontrolador digital que recebe informações de sensores e, com base em um script fornecido pelo usuário, pode reagir ao seu entorno, ligando e desligando luzes, regulando temperatura de ares-condicionados, abrindo e fechando cortinas, entre muitas outras possibilidades. O objetivo do projeto, de

acordo com o site oficial, é “criar ferramentas que são acessíveis, com baixo custo, flexíveis e fáceis de se usar por principiantes e profissionais”.

Com os Fab Labs e Arduinos, uma infinidade de projetos e plataformas vêm surgindo, preenchendo a lacuna entre o digital e o físico. A informação, na forma de código-fonte, sem metáforas, assume um papel protagonista, instigando o potencial das comunidades em código aberto.

Figura 7: Arduino Uno, um dos modelos mais acessíveis, voltado para prototipação.

Figura 8: Exemplo de script, transmitido para o Arduino via USB.



Fonte: https://pt.wikipedia.org/wiki/Arduino

Figura 9: Detahe de de movimento em intervenção artística urbana

Fonte: https://create.arduino.cc/projecthub/Maya/in-servo-we-trust-6725f1

Por parte dos arquitetos, um passo importante para a ignição dessas comunidades passaria pela atenção ao uso das leis de direitos autorais. Como percebido por Stallman no início do Projeto GNU, se não houvesse uma licença que garantisse a continuidade da liberdade do seu código, esse rapidamente seria aproveitado em softwares de código sigiloso. Ao adotar o uso de licenças *copyleft* o profissional pode impôr que as soluções por ele inventadas possam ser usadas livremente, mas que quaisquer modificações deverão ser igualmente compartilhadas sob a mesma licença, criando um efeito em cascata e um ambiente de colaboração, mesmo que pautado em interesses individuais. O ponto mais importante do uso correto das licenças não está na proteção de direitos de cópia, como ocorre tradicionalmente, mas sim no reconhecimento do autor, que deixa de ser "o proprietário único, hermético e inviolável de sua obra, mas sim o galho originário de uma árvore que brota galhos e botões com cada colaborador adicional" (CLAUDEL; RATTI, 2015, pág. 89).

Todo esse potencial colaborativo da arquitetura, porém, mesmo considerando o papel da informação após a obra construída, ainda não é, para Matthew Fuller e Usman Haque, suficiente para qualificar uma arquitetura em

código aberto por completo. Ao invés de enfatizar o compartilhamento da informação, os autores de *Urban Versioning System 1.0* defendem que arquitetura deveria superar a dualidade informação - matéria, buscando implementar um modelo de tentativa e -erro diretamente no espaço físico.

O manifesto parte de um questionamento importante, mas as estratégias apontadas têm viés utópico. A mais evidente se dá logo na primeira proposta, chamada “construa ao invés de projetar”, na qual os autores defendem que a concepção inicial deva ser o próprio edifício, desenhado no terreno escolhido.

> Construir continuamente, em vez de projetar, deixa claro que os edifícios são dinâmicos, responsivos e variáveis e incentivaria o desenvolvimento de estruturas tecnológicas robustas que unam projeto, construção e ocupação. (FULLER; HAQUE, 2008, tradução nossa)

Apesar de algumas provocações, há um ponto que é fundamental para o código aberto, tanto na informática, quanto na arquitetura: o estabelecimento de padrões universais, objetivando a compatibilidade de diferentes contribuições. Sem uma linguagem comum, compreendida e aceita por todos, uma construção colaborativa seria impossível. As linguagens de programação, por exemplo consistem em um conjunto de sintaxes que devem ser aceitas por todas as partes envolvidas no processo. A própria internet também é fruto de uma padronização de protocolos de comunicação bem sucedida, visto que quem discordar de tais protocolos e não desejar segui-los simplesmente estará fora da internet. É prudente que o projetista considere a questão, assim como fizera Stallman quando criou o GNU, buscando a compatibilidade com o sistema operacional Unix.

Na Era Digital, os elementos propícios para a emergência de uma arquitetura em código aberto se evidenciam: a internet, para o compartilhamento de informações e ideias, fomentando o surgimento de redes colaborativas em todo o mundo; mecanismos de proteção do autor, superando a ilegalidade da cópia, mas preservando o seu reconhecimento; a aproximação do digital e do real,

por meio de modelos cada vez mais próximos à realidade, equipamentos de fabricação digital, Arduinos; e por fim, a possibilidade de financiamento e engajamento coletivo, sendo uma alternativa ao modelo de negócios pautados nos interesses comerciais de incorporadoras ou interesses políticos. Mas como essas ideias vêm se manifestando na prática? Quais as perspectivas reais para uma arquitetura em código aberto?

3 DA TEORIA À PRÁTICA

3.1 APENAS COMPARTILHAR?

Atualmente, um dos primeiros resultados apresentados quando se pesquisa por arquitetura em código aberto é em referência ao escritório ELEMENTAL, de Alejandro Aravena. Famoso por uma abordagem participativa e por projetos de “moradia incremental”, projetando apenas “metade da casa” para que o usuário possa complementá-la de forma segura e personalizada, Aravena anunciou, logo após ganhar o Prêmio Pritzker 2016, que os projetos Quinta Monroy, Lo Barnechea, Monterrey e Villa Verde teriam toda a documentação, em PDF e também DWG, disponibilizada gratuitamente em seu site.

Figura 10: Quinta Monroy, Lo Barnechea, Monterrey e Villa Verde.

Fonte: https://www.archdaily.com.br/br/785050/elemental-disponibiliza-desenhos-de-4-projetos-habitacionais-para-uso-open-source

A abertura ao público desses arquivos, bem como a autorização para sua utilização e adaptação, gerou uma série de matérias e artigos sobre o tema. Em *Por que a iniciativa open source de Aravena é um grande passo para oferecer moradias melhores para todos*, publicado no ArchDaily em 2016, Joana Pacheco, uma das fundadoras da Paperhouses, diz:

> Ele, deliberadamente, renunciou aos direitos exclusivos de uso, o que significa ceder todos os lucros associados a execução destes desenhos. Fazendo isso, ele reconheceu que o papel mais importante destes projetos é inspirar outros a facilitar soluções aos problemas de moradia que enfrenta nosso mundo em sua rápida urbanização. (...) O último movimento atrevido de Aravena me dá ainda mais certeza de que o *open source* se tornará uma iniciativa comum que libertará o desenho e revolucionará a construção de moradias. (PACHECO, 2016)

A Paperhouses, por sua vez, foi um empreendimento que buscava disponibilizar projetos de qualidade, também gratuitamente, em seu site. Não se tratava de um escritório de arquitetura, eles não criavam casas, mas sim uma plataforma que recebia projetos detalhados em alto nível de outros arquitetos colaboradores. O argumento de Joana Pacheco para a iniciativa consiste na redução de custos do processo de concepção por meio do compartilhamento e da reutilização de soluções, o que é coerente com a crítica à "reinvenção da roda" de Kaspori (2003).

> Ignorando, por um momento, os problemas de escassez de material e segurança do abastecimento, (...) podemos ver que a mão-de-obra alcança aproximadamente um terço do custo de uma casa, enquanto os serviços (administrativos, financiamento, comercialização, arquitetura e engenharia) representam outro terço. Se aumentamos o papel da tecnologia e compartilharmos informações livremente sobre todas as partes de um projeto, desde o desenho até a fabricação, não somente podemos diminuir os custos de desenvolvimento da moradia, mas também melhorar sua qualidade. (PACHECO, 2016)

Figura 11: The Module House, projeto de Tatiana Bilbao S.C. para Paperhouses.

Fonte: https://www.archdaily.com.br/br/01-158111/paperhouses-arquitetura-de-codigo-aberto

Ambas as iniciativas tinham a pretenção de instigar novos compartilhamentos, promovendo uma mudança cultural entre os arquitetos e o sonho de um processo de concepção mais coletivo e evolutivo. No entanto, nenhuma dessas foi capaz de construir uma comunidade ativa de desenvolvimento ao seu redor. No caso dos projetos disponibilizados na Paperhouses, por exemplo, os arquitetos que acreditavam no ideal ofereciam os desenhos e esses eram no máximo adaptados para terrenos específicos, sem que passassem por aprimoramentos ou versões adicionais. O ciclo de vida dos projetos era finito, o foco era apenas a sua reprodução.

No caso das moradias incrementais, não se pode dizer que houve um fracasso absoluto, pois a publicação desses arquivos, de algum modo serviu para inspirar outros arquitetos e trazer visibilidade, tanto para sua abordagem projetual, quanto para seu próprio escritório. Mas ainda assim, apesar da abertura das informações arquitetônicas, tanto por parte de Aravena, quanto das

Paperhouses, nenhuma dessas alcançaram a magnitude, o impacto social e muito menos a organicidade, que um software livre pode ter.

O sucesso ou o fracasso de uma iniciativa não pode ser o único parâmetro para mensurar o potencial do conceito aqui debatido, mas também não pode ser ignorado. Para o arquiteto israelense Jonathan Dortheimer (2016), uma das razões para a ineficiência dessas iniciativas, quando comparadas à força do movimento código aberto na informática, se dá pelo simples fato de que o campo da arquitetura não apresenta um fechamento iminente, como ocorre entre os programadores. Ele observa que projetos arquitetônicos já são compartilhados com certa liberdade na internet por meio de sites como ArchDaily e Divisare, ou ainda em redes sociais e revistas especializadas. A apresentação pública desses projetos com plantas, cortes, fotografias e textos não só não são novidade na profissão, como constituem o principal elemento de ascensão individual da classe. Eles promovem o reconhecimento, a obtenção do capital intelectual, e com isso alimentam a cultura majoritária do campo, do arquiteto artista e autêntico, tão criticado por Claudel, Ratti e Kaspori.

Pode-se argumentar que os projetos apresentados no site do ELEMENTAL e da Paperhouses diferem dos projetos em redes sociais, por exemplo, porque os primeiros são compartilhados a nível executivo, com muito mais detalhes e transparência, enquanto no segundo caso, as informações são deliberadamente simplificadas e podadas, seja para facilitar a compreensão dos leitores, muitas vezes leigos, seja para não expor soluções particulares do escritório. De fato, há uma diferença considerável no formato, mas na prática ambos comunicam eficientemente o conceito e as soluções adotadas. Em alguns casos, inclusive, pode-se extrair maior aprendizado de projetos compartilhados em perfis em redes sociais, pois esses costumam vir acompanhados de textos explicativos do escritório, bem como uma maior interação do arquiteto, tirando dúvidas diretamente com seus seguidores.

A questão central é que, apesar das incontáveis variáveis com que os arquitetos precisam lidar para trazerem as melhores estratégias em cada contexto, as soluções arquitetônicas não costumam ter grande complexidade, podendo ser compreendidas com certa facilidade pela observação *in loco*. Enquanto o processo de concepção é, sim, extremamente complexo, os espaços edificados, quando comparados aos softwares, são simples e rapidamente aferidos. Em outas palavras, obras arquitetônicas permitem um processo de engenharia reversa pouco custoso, no qual se pode extrair as principais informações projetuais, ou, metaforicamente, o código-fonte, a partir da relação direta com o objeto.

Isso fica evidente quando se analisa as casas compartilhadas por Alejandro Aravena. Sua metologia de projeto para habitações sociais é admirável sob diversos aspectos, seja na viabilidade financeira, na facilidade de execução, ou na escuta ativa dos moradores, respeitando suas individualidades. Trata-se de um processo de muita sensibilidade, mas o produto em si, as casas construídas, podem ser compreendidas ao assistir sua palestra no TED Talks e analisar algumas fotos online.

Além da possibilidade de engenharia reversa, Jonathan Dortheimer, um dos poucos autores realmente críticos ao conceito, partindo da distinção entre conhecimentos explícitos e tácitos, proposta por Michael Polanyi em 1966, argumenta que não há muito potencial em uma arquitetura em código aberto porque a essência do conhecimento arquitetônico seria diferente daquele tipicamente compartilhado no campo da informática (2016). O conhecimento mais ligado aos softwares livres é o conhecimento explícito, ou seja, aquele que pode ser facilmente descrito e compartilhado por meio da linguagem, seja na forma de livros, manuais, infográficos, receitas, enfim. O código-fonte é um tipo de conhecimento explícito muito específico, sua linguagem não objetiva apenas a compreensão humana, mas também a comunicação direta com a máquina.

O conhecimento tácito, por sua vez, transcende a sistematização linguística, sendo muito mais difícil de ser transmitido. Polanyi (1966) acredita que algumas áreas, especialmente as ligadas à criatividade, são guiadas principalmente por esse conhecimento, relacionando-se mais às experiências adquiridas ao longo da vida, aos gostos pessoais e princípios morais, do que a um conjunto de informações registradas. Ambos os tipos de conhecimento são fundamentais em qualquer atividade humana, mas em arquitetura o verdadeiro diferencial competitivo, não está, segundo Dortheimer (2016), no conhecimento explícito, mas sim no conhecimento tácito de cada arquiteto.

Retomando o exemplo de Aravena, os desenhos em CAD por ele disponibilizados não abrangem a riqueza de sua produção, são apenas uma manifestação pontual e contextualizada dela. Seu discurso tem maior relevância que os arquivos compartilhados, não só porque suas obras podem ser compreendidas pela observação, mas porque, mesmo sem se tratar do conhecimento tácito, livros, palestras e entrevistas são uma melhor aproximação de sua experiência profissional.

Outro argumento para justificar a Paperhouses e as obras de Aravena como bons exemplos de arquitetura código aberto diz respeito ao uso das licenças de uso mais permissivas, o que não costuma ser cogitado em sites e revistas de arquitetura. Nas produções teóricas consultadas, licenças permissivas, geralmente do tipo *copyleft*, assumiram papel fundamental como viabilizador de uma arquitetura em código aberto. Claudel e Ratti, por exemplo, apontam a *Creative Commons* como instrumento central do desenvolvimento colaborativo, pois "enquanto a flexibilidade, a evolução e a adaptação continuam sendo possíveis, o poderoso motor da motivação humana, o reconhecimento da autoria, permanece intacto" (2015, pág. 90). Já o manifesto de Matthew Fuller e Usman Haque (2008) consiste, ele próprio, em uma protótipo de licença, que seria, segundo os autores, mais adequada para o “design e a construção de cidades em código aberto”.

Dortheimer (2016), porém, argumenta que as leis de proteção aos direitos autorais na arquitetura têm pouca ou nenhuma eficiência, pois elas não impedem que os profissionais reproduzam soluções de outros arquitetos. A cópia, em teoria, é considerada uma violação dos direitos autorais, mas na prática, apesar do campo ter forte influência das belas artes, tendo como premissa a originalidade, a reprodução de detalhes e soluções espaciais acaba fazendo parte do cotidiano da classe. Essas cópias, ou fortes inspirações, podem ser observadas tanto em tendências imobiliárias, atendendo a novas demandas ou modismos, quanto em adaptações tipológicas ou técnicas construtivas, de modo que a autorização ou a proibição se torna irrelevante.

Ainda, quando se propõe uma nova prática cujo pilar é o compartilhamento de informação, convém que esse se dê desde a fase inicial do projeto e acompanhe todo o seu ciclo de vida, em coerência com o lema “libere cedo, libere frequentemente”, cunhado por Raymond (1999), mas não é o que ocorreu nas iniciativas mencionadas. Em uma definição rasa, poderia-se considerar que os projetos das casas incrementais e as da Papherhouses são em código aberto, mas seu processo de desenvolvimento não diferiu em nada do modelo catedral, do modo tradicional de se projetar.

Buscando uma atuação mais alinhada ao movimento código aberto, o Opening Design, "um estúdio de arquitetura (absurdamente) transparente e de código aberto" (OPENING DESIGN, 2022, tradução nossa), sediado em Madison (EUA), implementa um processo de concepção aberto desde os primeiros momentos de cada projeto, compreendendo não só o compartilhamento de arquivos, mas um conjunto de estratégias que aumentam o potencial de colaboração por meio da internet. Em seu site, eles explicam:

> (...) todos os nossos desenhos, modelos, croquis, comunicação – tudo – são licenciados sob uma licença *copyleft*, em código aberto. Em um mundo altamente conectado, quando o conteúdo é gratuito e livre, ele pode evoluir de maneiras totalmente inesperadas e inovadoras. (...) essa abordagem demonstra que nosso valor como arquitetos e engenheiros

> não está no conjunto de desenhos que entregamos ao final do dia, mas sim no conhecimento e no processo necessário para chegar lá. (OPENING DESIGN, 2022, tradução nossa)

Importante ressaltar que a preocupação do escritório não se resume apenas a compartilhar, dispor na internet, mas também em adotar padrões universais, preferencialmente com uso de softwares livre, possibilitando que qualquer pessoa consiga ler e editar sem depender de programas específicos, muitas vezes pagos e de código fechado, como o Revit ou AutoCAD, por exemplo.

> Acreditamos que o uso de formatos proprietários (BIM/CAD) apenas dificulta a transmissão de informações. O conteúdo digital deve fluir sem impedimentos durante a duração de um projeto. Se um arquiteto, por exemplo, deseja compartilhar um arquivo BIM/CAD com um engenheiro estrutural que usa um software diferente, será impossível se os softwares não adotarem os padrões openBIM. Acreditamos que quando você impede aonde seu conteúdo pode ir, você impede com quem você pode trabalhar, o que acaba restringindo as opções e a realização de possíveis inovações e eficiências no futuro. (OPENING DESIGN, 2022, tradução nossa)

O campo da arquitetura, como visto, é instrinsecamente aberto, suas informações projetuais não são difíceis de obter, seja pela observação das obras edificadas, seja pela necessidade de autopromoção dos profissionais que inevitavelmente compartilham seus projetos online. O Opening Design leva essa abertura um passo além, compartilhando as etapas iniciais de cada projeto, até mesmo as conversas da equipe. Ao contrário das primeiras iniciativas tratadas, o escritório apresenta uma visão muito consciente do conceito de código aberto, buscando projetar com a participação de pessoas em todo o mundo por meio da internet. No entanto, ainda não se pode dizer que em seus projetos configuram-se comunidades de desenvolvimento semelhantes ao modelo bazar. Apesar do esforço e de certa esperança para atrair colaboradores, as pessoas envolvidas nos projetos são apenas parte da equipe contratada, sejam membros do próprio

escritório, sejam especialistas consultados (DORTHEIMER, 2016, pág. 76, tradução nossa). Mas, então, o que falta para a efetivação de uma arquitetura em código aberto?

3.2 UM NOVO PARADIGMA?

Devido ao sucesso de suas práticas e também à força dos seus ideais, alinhados a movimentos progressistas e à contracultura, o código aberto gerou interesse de teóricos e projetistas que buscavam uma produção arquitetônica mais colaborativa, eficiente e democrática. Desenvolver em código aberto soava como uma dádiva, capaz de responder a quaisquer demandas sociais. Mas, como pôde-se perceber, não se trata de um conceito tão irrestrito, aplicável sem ressalvas em qualquer campo produtivo.

O professor de direito Yochai Benkler, famoso por seus estudos sobre economia na sociedade em rede, costuma mencionar o código aberto como um exemplo prático de um modelo mais amplo, por ele definido como *common-based peer production*, ou, comumente traduzido, produção colaborativa baseada em recursos comuns. Entende-se como *commons* os bens que são compartilhados por uma comunidade, como o ar ou mesmo uma via pública, mas no contexto da produção baseada em recursos comuns, ele é associado a bens informacionais ou digitais.

> A produção colaborativa baseada em recursos comuns é um sistema socioeconômico de produção que está surgindo no ambiente digitalmente conectado em rede. Facilitado pela infraestrutura técnica da internet, a marca desse sistema sociotécnico é a colaboração entre grandes grupos de indivíduos, às vezes dezenas ou mesmo centenas de milhares, que cooperam efetivamente para fornecer informações, conhecimentos ou bens culturais sem depender de preços de mercado ou hierarquias gerenciais para coordenar o empreendimento comunitário. (BENKLER, 2006, tradução nossa)

Quando Kaspori e Raymond argumentam que para fomentar o surgimento de comunidades em código aberto é necessário um interesse compartilhado, na verdade eles estão apontando que, tanto na arquitetura, quanto na computação, as comunidades surgem em torno de *commons*. No campo da informática, os softwares são naturalmente um recurso comum, pois um único código-fonte é usado em todo o mundo, com propósitos distintos. Um aplicativo de calculadora, por exemplo, consiste em apenas algumas linhas de código, mas pode servir a tantos fins que seria inútil tentar mensurar. Se o aplicativo for um software livre, a adição de um novo recurso, realizada por um único usuário, impactará diretamente no uso de milhares ou mesmo milhões de outros usuários que, por sua vez, também poderão adicionar novos recursos.

Esse fato ajuda a entender o porquê de nenhuma das iniciativas de código aberto discutidas até aqui terem conseguido fomentar essas comunidades. Mesmo sob o argumento de que os projetos compartilhados podem ser reutilizados ou adaptados noutros terrenos, as soluções são essenciamente locais, para um público relativamente pequeno e pré-estabelecido. O escopo da profissão do arquiteto pode ser abrangente, mas a sua atuação está majoritariamente direcionada ao desenvolvimento de soluções em um lugar específico, tendo como parâmetros o clima, a cultura, materiais e mão de obra disponíveis, entre outros, sem esquecer das demandas individuais de cada cliente.

Entende-se, portanto, que o potencial de uma arquitetura em código aberto não está tanto na concepção de projetos para um contexto local, mas sim no desenvolvimento de soluções que possam ser compreendidas como *commons* por outros arquitetos e até autoconstrutores. Desse modo, existe, sim, uma possibilidade para a implementação do modelo bazar na arquitetura, mas essa não implica em uma reconfiguração cultural e estrutural do campo, um novo paradigma de produção, como apresentado na definição de Carlo Ratti, ou uma “reviravolta no pensamento sobre os princípios organizacionais fundamentais da prática arquitetônica”, como almejou Kaspori em 2003.

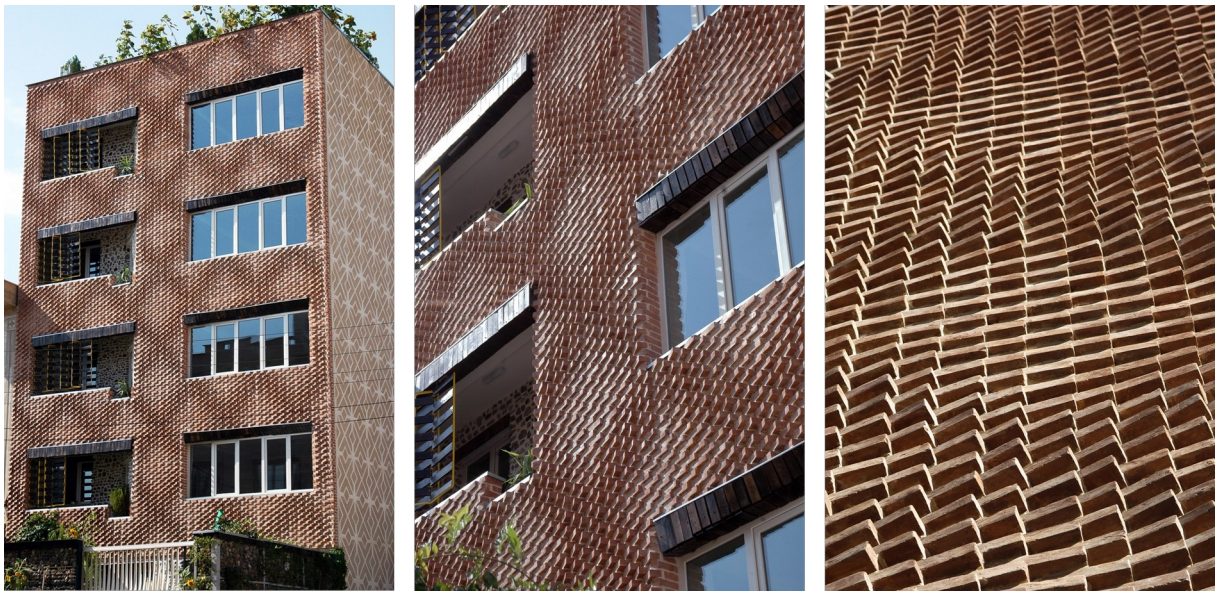
De certa forma, tipologias e técnicas construtivas podem ser consideradas recursos comuns, pois tratam-se de conhecimentos compartilhados entre uma população, sendo por ela aprimorados ao longo de gerações. Mas como visto anteriormente, esse tipo de informação já apresenta um grau de abertura satisfatório, de modo que o compartilhamento ou não de informações projetuais não trará nenhum impacto considerável. Com isso, tem-se uma outra condição para justificar uma arquitetura em código aberto: a informação deliberadamente compartilhada deve ser fundamental para a compreensão do produto a ser desenvolvido, ou seja, a possibilidade de engenharia reversa deve ser mínima. Não se trata de um ideal a ser atingido pela produção arquitetônica, mas, simplesmente, um contexto em que ela pode ser mais relevante ou até necessária.

Em síntese, o impacto da informação compartilhada em código aberto é proporcional ao seu potencial de uso em diferentes contextos e à dificuldade de compreensão do seu funcionamento por meio de engenharia reversa.

Cumprindo esses dois critérios, um projeto interessante é o site Bricksource, mantido pelo escritório Sstudioumm. Após criarem um método simples, *low-tech*, de concepção e construção de fachadas de alvenaria com padrões parametrizados, exigindo apenas um conjunto de estênceis ajustáveis, inventados pelo próprio escritório, e um arquivo contendo a angulação de cada tijolo, os autores decidiram compartilhar, não só as instruções para confecção desses estênceis, mas também os padrões por eles utilizados.

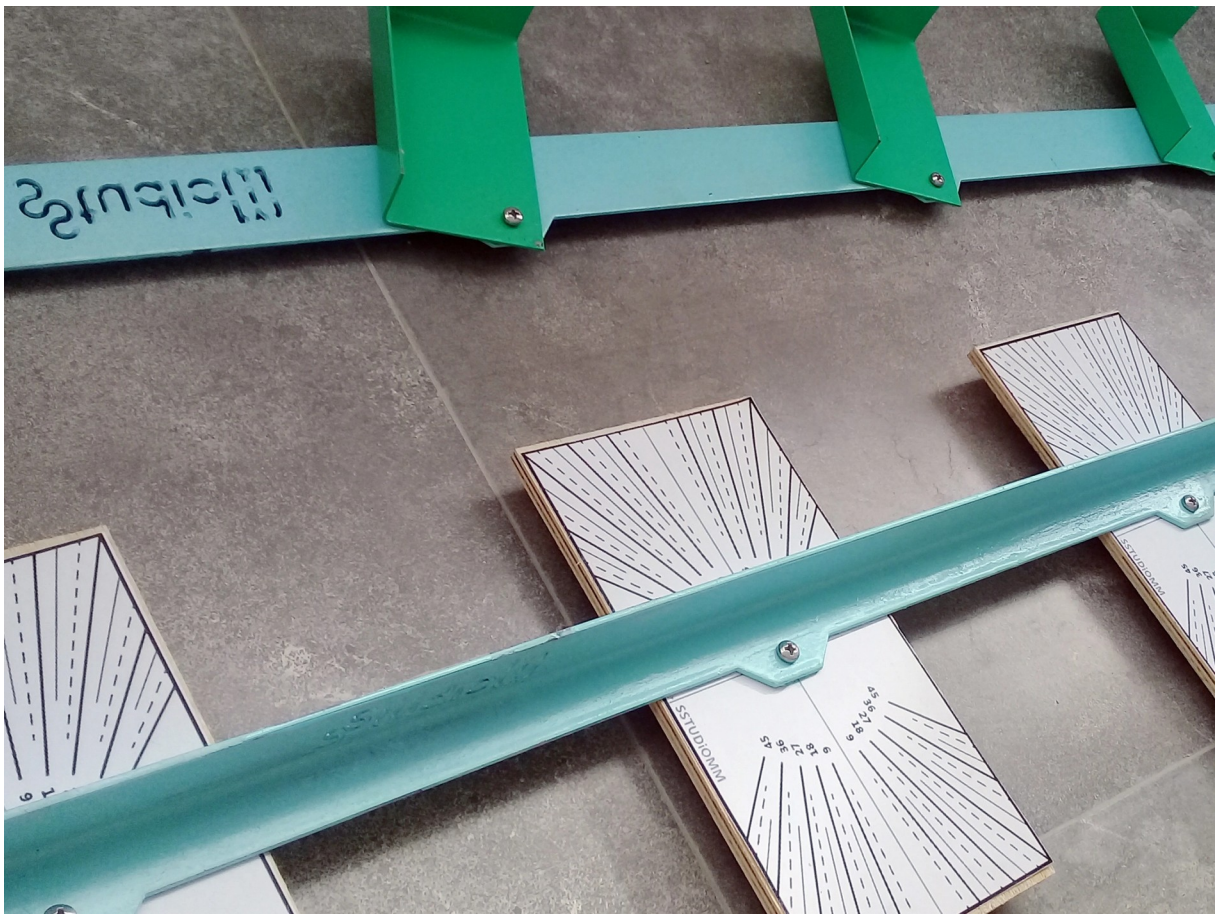
Esse método foi desenvolvido para ser aplicado em projetos no Irã, mas ao perceberem seu potencial em outros edifícios, os autores optaram por compartilhar o conhecimento. Cada padrão foi disponibilizado em três formatos: em PDF, em planilhas de Excel, e em *scritpts* do GrassHopper, um *plugin* de design paramétrico para o Rhinoceros. Os dois primeiros são apenas um tipo de desenho executivo, e, mesmo que não faça sentido, poderiam ser aferidos no lugar. Já o terceiro, o mais valioso, permite compreender suas regras compositivas, não apenas para reproduzir, mas também para editar e ajustar.

Figura 12: APT NO 7, fachada projetada pelo Sstudiumm.



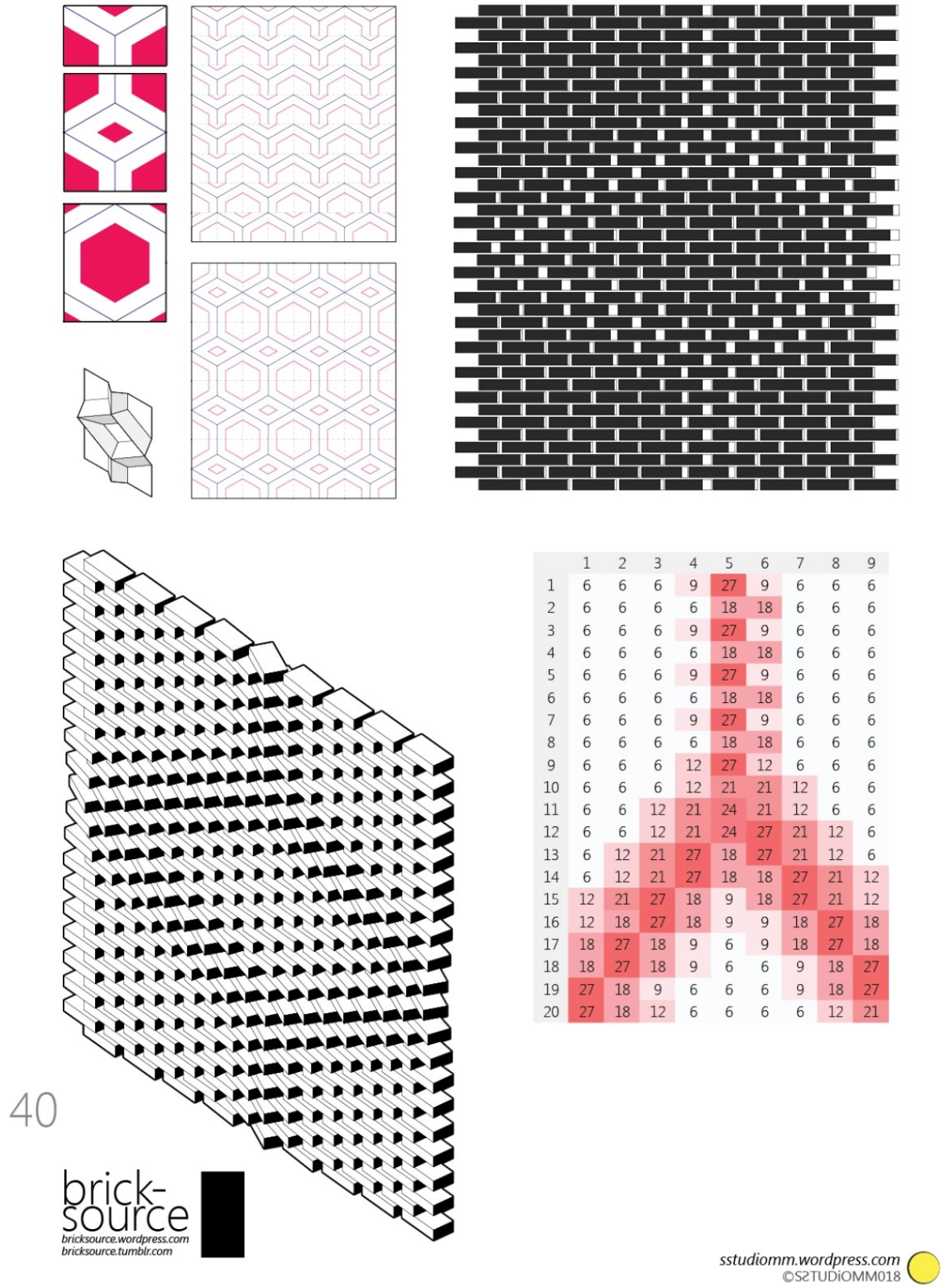
Fonte: https://bricksource.wordpress.com/2017/12/26/apt-no-7-brick-weaving/

Figura 13: dois tipos de estênceis ajustáveis, desenvolvidos pelo Sstudiumm.



Fonte: https://sstudiomm.wordpress.com/2017/12/26/adjustable-stencils/

Figura 14: exemplo de padrão compartilhado no Bricksouce.



	1	2	3	4	5	6	7	8	9
1	6	6	6	9	27	9	6	6	6
2	6	6	6	6	18	18	6	6	6
3	6	6	6	9	27	9	6	6	6
4	6	6	6	6	18	18	6	6	6
5	6	6	6	9	27	9	6	6	6
6	6	6	6	6	18	18	6	6	6
7	6	6	6	9	27	9	6	6	6
8	6	6	6	6	18	18	6	6	6
9	6	6	6	12	27	12	6	6	6
10	6	6	6	12	21	21	12	6	6
11	6	6	12	21	24	21	12	6	6
12	6	6	12	21	24	27	21	12	6
13	6	12	21	27	18	27	21	12	6
14	6	12	21	27	18	18	27	21	12
15	12	21	27	18	9	18	27	21	12
16	12	18	27	18	9	9	18	27	18
17	18	27	18	9	6	9	18	27	18
18	18	27	18	9	6	6	9	18	27
19	27	18	9	6	6	6	9	18	27
20	27	18	12	6	6	6	6	12	21

Fonte: https://sstudiomm.wordpress.com/2017/12/26/adjustable-stencils/

Sobre os três formatos compartilhados pelo escritório, embora todos contenham informações relevantes para a execução, os *scripts* são os que mais se assemelham ao conceito de código aberto na arquitetura. As angulações disponibilizadas em PDF ou em Excel são importantes do ponto de vista didático, facilitando a utilização por pessoas com diferentes níveis de experiência, mas na verdade elas são apenas um produto dos próprios *scripts*. No contexto dessa técnica específica, o compartilhamento dos códigos é absolutamente essencial, pois sem ele o construtor teria que recriá-los do zero.

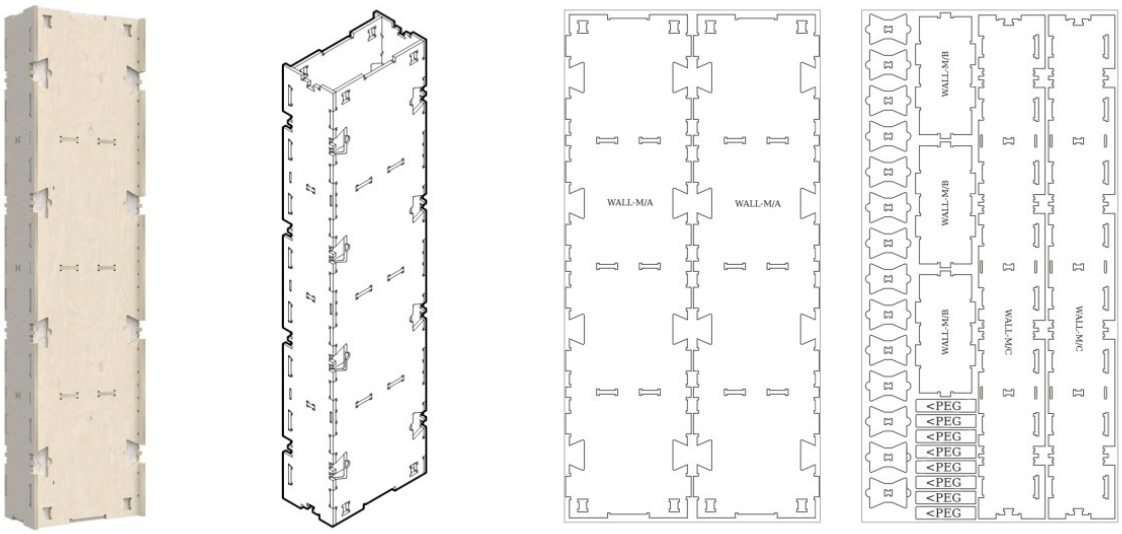
Assim, além do fato desses padrões serem recursos comuns, passíveis de utilização e adaptação em diferentes contextos, justifica-se a ideia de código aberto também porque, ao contrário das técnicas construtivas tradicionais, se não houver um compartilhamento consciente por parte dos autores, tal técnica estaria literalmente em código fechado. Quando não há a possibilidade de fechamento da informação arquitetônica, a ideia de uma arquitetura em código aberto se esvazia.

A Bricksource é um iniciativa pequena, atendo-se apenas a uma técnica de composição de fachadas, não se apresentando como catalizador de uma revolução cultural e estrutural do campo, como a Paperhouses. Há, porém, um outro projeto que apresenta um espectro de uso muito mais amplo, considerado por Dortheimer (2016) como o único empreendimento de arquitetura em código aberto bem sucedido: a WikiHouse. Vale lembrar que a Bricksource foi lançada aproximadamente 1 ano após a publicação de sua tese.

Apresentada ao público em 2011 por Alastair Parvin e Nick Ierodiaconou 00, estando atualmente na versão Skylark, a WikiHouse surgiu como uma alternativa mais eficiente e ecológica do que a alvenaria tradicional, e mais acessível e democrática do que os sistemas de construção pré-fabricados. Sua missão é “colocar nas mãos de todos os cidadãos, comunidades e empresas o conhecimento e as ferramentas necessárias para a contrução de edifícios bonitos e zero-carbono” (WIKIHOUSE, 2022, tradução nossa). Segundo o FAQ do projeto:

> O WikiHouse é um sistema de construção fabricado para casas (na verdade, pode ser usado para muitos tipos de pequenas construções). Ele usa folhas de madeira estrutural (geralmente madeira compensada) que são cortadas com precisão de 0,1 mm e montadas em blocos de construção básicos, que podem ser entregues no local e montados com rapidez e precisão por quase qualquer pessoa, mesmo que não tenham habilidades tradicionais de construção. (WIKIHOUSE, 2022, tradução nossa)

Figura 15: WALL M, módulo de parede tamanho médio, para pé-direito de 2.40m.



Fonte: https://www.wikihouse.cc/blocks/wall-m

Por ser um sistema construtivo versátil e com muitas vantagens competitivas, a Wikihouse conseguiu um relativo sucesso, fomentando o surgimento de comunidades de arquitetos atentos a melhorar o sistema construtivo, e também a criar adaptações para determinados climas. Cada alteração desenvolvida para um projeto, pode ser acrescida à biblioteca oficial, seguindo as orientações do site:

> Se você fez melhorias em um bloco Skylark, desenvolveu novos blocos ou até mesmo um produto totalmente novo que você acha que podemos querer mesclar na biblioteca principal, você pode criar um *fork* no Github ou apenas enviar seus arquivos para nós, juntamente com uma descrição. (WIKIHOUSE, 2022, tradução nossa)

Figura 16: Casa Revista, primeira WikiHouse do Brasil. Projeto de Clarice Rohde.

Fonte: https://www.archdaily.com.br/br/773676/casa-revista-a-primeira-casa-fabricada-digitalmente-no-brasil

Figura 17: Casa LG ThinQ, definida como *open source* pelos autores, mas seu projeto não é público.

Fonte: https://www.archdaily.com.br/br/971831/casa-lg-thinq-estudio-guto-requena-plus-pax-arquitetura

Uma crítica necessária se dá pelo fato de que os blocos disponibilizados estão em 3 formatos proprietários, 3dm, skp e dwg, padrões apenas de softwares pagos, quando deveriam vir também em formatos abertos, preferencialmente de acordo as diretrizes da openBIM, como faz o escritório Open Building. O uso de formatos proprietários, em si, não é um erro, mas não poderia ser a única opção.

Outro ponto questionável consiste na não obrigatoriedade de compartilhamento dos projetos usando a Wikihouse. Mesmo que soluções locais não sejam tão relevantes para outros usuários quanto o sistema em si, adaptações locais ainda teriam valor, visto que a Wikihouse é um sistema mais complexo e recente, ainda em fase de experimentação. Perde-se uma oportunidade, obviamente, mas trata-se de uma decisão estratégica compreensível, não descaracterizando-a enquanto sistema construtivo em código aberto.

> Como princípio geral, nossa abordagem é que o sistema de construção WikiHouse é nosso para publicar, mas o layout e a especificação de um projeto específico pertencem dos criadores desse projeto, e cabe a eles publicar esses arquivos ou não. Na prática, desenhos para projetos específicos são menos úteis do que você imagina, pois são muitas as características específicas daquele lugar e época. (WIKIHOUSE, 2022, tradução nossa)

Inspirados no aspecto comunitário do campo da informática, especialmente no modelo de desenvolvimento de softwares em código aberto, muitos arquitetos e teóricos vislumbraram uma produção arquitetônica também em código aberto, mais colaborativa, justa e democrática. O modelo bazar apresentava-se como um ideal e, com a integração das tecnologias digitais, a arquitetura em código aberto parecia cada vez mais plausível.

A noção de código aberto, no entanto, surge a partir de um movimento social de resistência, em oposição a um processo de supressão e sufocamento de comunidades colaborativas. Richard Stallman fundou o movimento software livre

em resposta a uma ameaça e, apenas anos depois, o potencial do modelo bazar foi percebido e estudado.

Apesar da busca por um antagonista no campo da arquitetura, a exemplo de Le Corbusier e dos arquitetos estrelas, não há um sentimento de revolta evidente entre os arquitetos, como ocorreu entre os hackers do MIT. Mesmo que o isolamento descrito por Ratti e Kaspori possa ser compreendido como uma perda de espaço, assim como sofrera Stallman, na prática os arquitetos parecem estar confortáveis com essa estrutura, pois dela se beneficiam. Em outras palavras, enquanto na informática os softwares livres surgem como uma reação a uma força oposta, a arquitetura em código aberto surge a partir de um questionamento sobre o que o campo poderia ser e significar para o mundo.

A arquitetura, considerando o trabalho dos profissionais e também dos autoconstrutores, como observado por Claudel e Ratti (2015), apresenta características semelhantes ao bazar de Raymond, de modo que, uma arquitetura em código aberto seria apenas uma manutenção desse modelo, mas potencializada e expandida pelas tecnologias de informação e de comunicação. Assim, a princípio, o código aberto não poderia resultar em um novo paradigma de produção do espaço, pois o conhecimento arquitetônico não apenas flui com liberdade, como é fundamental para a própria estrutura de incentivo do campo. Do mesmo modo, a discussão sobre código aberto também não tinha relevância a dez ou quinze anos antes da fundação do movimento software livre, pois o compartilhamento dos códigos era a regra do jogo.

Com a crescente integração do digital na arquitetura, otimizando processos de concepção e produção e ainda criando novas possibilidades de interação com o espaço edificado, cresce também uma ameaça à natureza aberta do campo, assim como ocorreu na informática. As tecnologias digitais apresentam o argumento da eficiência, do custo, da rapidez, da personalização, da sustentabilidade, possivelmente amenizando questões habitacionais e ambientais. Mas esses benefícios acompanham uma complexificação das informações relativas ao

projeto, tornando-os inacessíveis. O que antes seria uma planta, pode se tornar um conjunto de arquivos, algoritmos, sistemas construtivos indecifráveis ou protegidos por patentes, por exemplo. Isso excluiria a possibilidade de participação do usuário e, em casos extremos, ele seria impedido de reformar sua própria casa. A experiência arquitetônica se resumiria a uma relação produto-usuário, como acontece em tantos outros setores produtivos.

Em 2018, uma nova startup, a Icon, nasceu nos Estados Unidos com a missão de construir habitações emergenciais usando máquinas de impressão 3D. A primeira casa foi construída no Texas e dois anos depois, novas unidades foram produzidas em série para pessoas desabrigados. Assim como a WikiHouse, a empresa também adota o discurso da eficiência, ressaltando o tempo reduzido de construção e, consequentemente, um menor custo com mão de obra:

> A impressora 3D de construção em larga escala é 1,5x maior, 2x mais rápida e capaz de imprimir casas e estruturas de até 3.000 pés quadrados. O Vulcan pode produzir casas térreas resilientes mais rapidamente do que os métodos convencionais e com menos desperdício e mais liberdade de design. (ICON, 2022, tradução nossa)

O material utilizado pelas impressoras 3D, desenvolvidas pela própria empresa, é chamado de Lavacrete, e sua fórmula é patenteada e sigilosa. Segundo a Icon, o Lavacrete é composto de materiais comuns e seu preparo ocorre em pequenas unidades de produção automatizadas, as Magmas, anexadas diretamente às impressoras, dispensando, ou evitando, a manipulação e o consequente vazamento da receita.

> Usando qualquer uma das misturas de Lavacrete proprietárias da ICON, o sistema Magma mistura Lavacrete, aditivos e água automaticamente, dependendo das condições climáticas atuais do local, e então fornece o Lavacrete pronto para impressão ao Vulcan. Inteligente e trabalhando em perfeita sincronia com uma impressora Vulcan no local, a Magma elimina as suposições de materiais cimentícios complexos e de alto desempenho. (ICON, 2022, tradução nossa)

Figura 18: Impressora 3D da Icon.

Fonte: https://ciclovivo.com.br/arq-urb/arquitetura/vila-casas-impressas-em-3d-sem-tetos/

O sistema construtivo da Vulcan, bem como o modelo de negócios da startup, mesmo que igualmente fundamentado nas tecnologias de fabricação digital, é completamente antagônico ao sistema da WikiHouse. Enquanto a WikiHouse compartilha cada detalhe dos seus módulos, visando que qualquer pessoa em qualquer lugar possa projetá-la, configurando uma comunidade de usuários e colaboradores espalhada pelo mundo, a Icon assume uma posição centralizadora, da concepção até a produção das casas, dificultando deliberadamente a troca de informações. Se a Wikihouse é um exemplo de arquitetura em código aberto, o Vulcan é a sua antítese.

Atualmente, a ideia de fechamento pode ainda não ser tão evidente, pode não estar muito clara ou não incomodar tanto. O volume de construções baseadas em tecnologias de fabricação digital, por exemplo, ainda é irrelevante perto da produção vernacular desenvolvida e aprimorada ao longo de séculos. No entanto,

o avanço dessas tecnologias em todos os campos produtivos é inevitável e surpreendentemente rápido. Nesse contexto, mais do que uma alternativa mais eficiente à alvenaria tradicional, a WikiHouse é o princípio de uma resistência a sistemas construtivos e modelos de negócios que tolhem o compartilhamento de conhecimentos, de maneira análoga às empresas que provocaram a revolta de Stallman.

Não se objetiva a substituição de técnicas construtivas tradicionais pelo uso de tecnologias de ponta, mas, assumindo que a popularização do digital é um caminho sem volta, é importante que os arquitetos disponham de soluções consolidadas, de alta performance, e em código aberto.

CONCLUSÃO

Na informática, o discurso do movimento software livre sempre esteve associado ao seu oposto, aos softwares em código fechado. Não há código aberto quando todos são abertos.

Na arquitetura esse antagonismo não é tão presente, nem nas discussões teóricas, nem nas iniciativas práticas, visto que o campo da arquitetura não apresenta ainda um tendência de fechamento tão iminente quanto na computação. Mesmo que o compartilhamento de plantas e cortes tenha sido interpretado como princípio de um "novo paradigma", nenhum dos autores arriscou classificar um projeto ou empreendimento como representante da arquitetura em código fechado.

A descrição de um vilão, o arquiteto artista e individualista, de fato ocorreu em alguns trabalhos teóricos, mas a presente pesquisa demonstra que esses arquitetetos e os hackers que fundaram o movimento software livre, respondem a uma estrutura de incentivos semelhante, de modo que, mesmo que a crítica seja bem fundamentada, ela não tem uma forte relação com o movimento código aberto. O discurso de boas novas do modelo bazar fez brilhar os olhos, enquanto as razões que fundamentaram o surgimento do software livre conceito foram deixadas de lado.

A pesquisa, porém, trás um outro ponto de vista: a possibilidade de uma arquitetura em código fechado, com origem justamente nas tecnologias digitais. Explorar as tecnologias digitais na arquitetura, portanto, não deve ser apenas uma possibilidade projetual, mas uma busca pelo desenvolvimento de soluções de alta performance que, se não forem criadas logo, serão tomadas por outras mais eficientes, poderosas e em código fechado. É preciso pensar a arquitetura em código aberto não apenas como um caminho mais eficiente, mas principalmente como um movimento social.

REFERÊNCIAS

CARDOSO, C. A. P. Formas Arquitetônicas em Ambiente Computacional. *In*: VIII CONGRESSO IBERO-AMERICANO DE GRÁFICA DIGITAL, 2004, São Leopoldo. **Proceedings...** Porto Alegre: Unisinos, 2004. p 317-319.

DORTHEIMER, J. **Open Source Architecture**: Challenges and opportunities. Tel Aviv University, 2016.

FULLER, M; HAQUE, U. **Uban Versionirng System 1.0**. Lulu.com, 2008.

GALERIA DA ARQUITETURA. Criatura de Luz. Disponível em: https://www.galeriadaarquitetura.com.br/projeto/estudio-guto-requena_/wz-hotel/1647. Acesso em: 2 abr. 2022.

GUTO REQUENA. Bar Emotividade. Disponível em: https://gutorequena.com/baremotividade. Acesso em: 2 abr. 2022.

HABRAKEN, N. J.; TEICHER, J. **Structure of the ordinary**: form and control in the built environment. Cambridge Ma.: Mit Press, 1998.

HABRAKEN, N. John. Questions That Will Not Go Away: Some Remarks on Long-Term Trends in Architecture and their Impact on Architectural Education. **Open House International**, v. 31, n. 2, p. 12–19, 1 jun. 2006.

HIMANEN, P. **A Ética dos Hackers e o Espírito da Era da Informação**. Rio de Janeiro: Campus, 2001.

ICON BUILD. Disponível em: https://www.iconbuild.com. Acesso em: 6 mar. 2022.

KASPORI, D. **A Communism of ideas**: Towards an open-source architectural practice. 2003. Disponível em: https://files.stample.co/stample-1464627115781-ACommunismOfIdeas.pdf. Acesso em: 8 mar. 2022.

LEVY, P. **Cibercultura**. São Paulo (Sp): Ed. 34, 1999.

MELO, A. P. de et al . O conhecimento tácito a partir da perspectiva de Michael Polanyi. **Arq. bras. psicol.**, Rio de Janeiro , v. 71, n. 2, p. 34-50, 2019 . Disponível em: http://pepsic.bvsalud.org/scielo.php?script=sci_arttext&pid=S1809-52672019000200004. Acesso em 12 mar. 2022.

NASCIMENTO, D. M.. Uma leitura bourdieusiana da arquitetura. In: MARTELETO, R. M.; PIMENTA, R. M. **Pierre Bourdieu e a produção social da cultura, do conhecimento e da informação**. Rio de Janeiro: Garamond, 2017.

NEGROPONTE, N.. **Soft Architecture Machines.** Cambridge, Mass. / London: Mit Press, 1975.

PEREIRA, I.. **O Movimento do software livre**. In: Congresso Luso-Afro Brasileiro de Ciências Sociais. Coimbra, 2004.

Open-source architecture. *In*: WIKIPEDIA: a enciclopédia livre. Disponível em: https://en.wikipedia.org/wiki/Open-source_architecture. Acesso em: 30 mar. 2022.

RATTI, C.; CLAUDEL, M. **Open Source Architecture**. Nova Iorque: Thames & Hudson, 2015.

RAYMOND, E. S. **A catedral e o bazar**. 1998. Disponível em: https://www.ufrgs.br/soft-livre-edu/arquivos/a-catedral-e-o-bazar-eric-raymond.pdf. Acesso em: 8 mar. 2022.

REQUENA, C. A. J. **Habitar híbrido**: subjetividades e arquitetura do lar na era digital. Editora Senac São Paulo, São Paulo, 2019.

REVOLUTION OS. Direção de J. T. S. Moore. Wonderview Productions. 2001

SANCHES, W. D. **O Movimento de Software Livre e a Produção Colaborativa do Conhecimento**. Pontifícia Universidade Católica, São Paulo, 2007.

SENAGALA, M. **Speed and Relativity: Toward Time-like Architecture**. University of Texas, San Antonio, s.d.

SILVEIRA, S. A. **Ciberativismo, cultura hacker e o individualismo colaborativo**. Revista USP, v. 0, n. 86, p. 28, 1 ago. 2010.

SILVEIRA, S. A. **Sofware livre:** a luta pela liberdade do conhecimento. São Paulo: Fundação Perseu Abramo, 2004.

SOUSA, M. Vila de casas impressas em 3D é construída para sem-tetos. Ciclo Vivo, 2021. Disponível em: https://ciclovivo.com.br/arq-urb/arquitetura/vila-casas-impressas-em-3d-sem-tetos/. Acesso em: 9 abr. 2022.

SSTUDIOMM. Adjustable Stencils. Disponível em: https://sstudiomm.wordpress.com/2017/12/26/adjustable-stencils/. Acesso em: 4 abr. 2022.

STALLMAN, R. **Unix Livre!** 1983. Disponível em: https://www.gnu.org/gnu/initial-announcement.pt-br.html. Acesso em: 8 mar. 2022.

STOTT, R. 5 Iniciativas que mostram a ascensão da arquitetura open source. ArchDaily, 2016. Disponível em: https://www.archdaily.com.br/br/797398/5-iniciativas-que-mostram-a-ascensao-da-arquitetura-open-source. Acesso em: 7 abr. 2022.

TOLLSTADIUS, L. WikiHouse: uma casa para imprimir. B9, 2012. Disponível em: https://www.b9.com.br/30161/wikihouse-uma-casa-para-imprimir/. Acesso em: 4 abr. 2022.

TORVALDS, L.; DIAMOND, D. **Só por prazer, Linux**: os bastidores da sua criação. Rio De Janeiro: Campus, 2001.

USINA FAB LAB. O que é um Fab Lab? Disponível em: https://www.usinafablab.com.br/fablab. Acesso em: 6 mar. 2022.