MIT Libraries | DSpace@MIT

MIT Open Access Articles

Open Source Architecture: An Exploration of Source Code and Access in Architectural Design

The MIT Faculty has made this article openly available. ***Please share*** how this access benefits you. Your story matters.

Citation: Vardouli, Theodora, and Leah Buechley. "Open Source Architecture: An Exploration of Source Code and Access in Architectural Design." Leonardo 47, no. 1 (February 2014): 51–55. © 2014 ISAST

As Published: http://dx.doi.org/10.1162/LEON_a_00470

Publisher: MIT Press

Persistent URL: http://hdl.handle.net/1721.1/85899

Version: Final published version: final published article, as it appeared in a journal, conference proceedings, or other formally published context

Terms of Use: Article is made available in accordance with the publisher's policy and may be subject to US copyright law. Please refer to the publisher's site for terms of use.

Massachusetts Institute of Technology

Open Source Architecture: An Exploration of Source Code and Access in Architectural Design

Theodora Vardouli and Leah Buechley

ABSTRACT

The term *open source* is increasingly applied to architecture, yet there is little consensus about what it means in this context. This paper explores how different literal and metaphoric interpretations of the "access to source code" principle, set by the founders of the Free and Open Source Software movements, are being applied to architecture. The authors explore several challenges that have arisen in the translation of open source rhetoric from cyberspace to architectural space and discuss paths for new conceptual and programmatic agendas promoting user empowerment and democratization in architectural design.

OPEN SOURCE EVERYTHING

Increasing technological literacy, the popularization of hacking and do-it-yourself (DIY) and the growth of creative online communities have destabilized traditional models of design. The notion of "access" (to information, tools, designs, etc.) and models of production based on networks of collaborating individuals have become central discursive axes in diverse fields of human activity. These discussions are pragmatic, yet also vested with a utopian character, linking rhetoric of "democratization" and user empowerment to visions of decentralization and personal creativity.

Free/Libre, Open Source Software (FLOSS) [1] exemplifies both a new model of production and a social vision [2], building on the emancipatory potential of non-hierarchical and egalitarian production where individuals and collectives can access, modify and distribute the technologies they utilize. Denoting both a pragmatic organizational model and an ideological position, the ideas and practices of FLOSS have now transcended the world of software and are gaining ground in the collective imaginary. The alleged Linus Torvalds quote "The future is open source everything" [3] has become the impetus for a common prospective endeavor.

The growing wave of translations and interpretations of the tools, practices and concepts of FLOSS across different domains illustrates the evocative power of the phrase *open source* and its positive connotations as a brand. In most appropriations, the phrase drifts away from its initial meaning and functions as a metaphor that is used either to label existing practices or to motivate explorations of new ones. According to social scientist Dale Bradley, the cross-disciplinary use of *open source* operates primarily at a symbolic level. He argues that the phrase is used as a synonym for "open," to suggest a more horizontal, inclusive and participatory approach to design and production:

> It is the anarcho-utopian element of FLOSS that is most frequently cited as a model to be emulated and/or adapted to broader social formations and practices. FLOSS is therefore as much about anarcho-utopianism as it is about programming, because what marks FLOSS as different from traditional software development is not a new technical practice—coding languages remain largely unchanged—but a new social practice of software production, distribution, and use [4].

Free Software pioneer Richard Stallman strongly criticizes this metaphoric use. He argues that the FLOSS ideology and methodology are software-specific:

> The term "open source" has been further stretched by its application to other activities, such as government, education, and science, where there is no such thing as source code, and where criteria for software licensing are simply not pertinent. The only thing these activities have in common is that they somehow invite people to participate. They stretch the term so far that it only means "participatory" [5].

Stallman's call for caution is inscribed within his broader criticism of popular misunderstandings of the phrase *open source,* which obscure its ideological orientation and specificity of practices [6]. The first cause of this confusion is the obfuscation of the philosophical differences between "free" and "open source." Although the words are often popularly assumed to be synonymous or interchangeable, there is a distinct difference between the socialist orientation of Richard Stallman's Free Software and the libertarian approach of Eric Raymond's Open Source Initiative (OSI) [7]. Stallman argues that the freedom to access, share and modify software's source code should be a fundamental human right. On the other hand, Raymond and the OSI adopt a utilitarian, business-oriented approach, arguing that open source practices enable faster and better software development [8]. This distinction is often overlooked as open source rhetoric migrates to different domains. In these cross-disciplinary appropriations, the term is used loosely to communicate a desire for openness, collaboration and participation.

As open source rhetoric informs and inspires new cultures of production and use [9], it is important to question when and where its ideas and practices can be reasonably applied. As Stallman laments, in many cases the phrase *open source* is used opportunistically for its positive cultural connotations, with little consideration of its actual meaning. In particular, the absence of source code—or an analogous entity—in many

Theodora Vardouli (researcher), Design and Computation, Department of Architecture, Massachusetts Institute of Technology, 77 Massachusetts Avenue, Cambridge, MA 02139-4307, U.S.A. E-mail: <thvard@mit.edu>.

Leah Buechley (educator), Massachusetts Institute of Technology, Department of Media Arts and Sciences (Media Lab), 77 Massachusetts Avenue, Cambridge, MA 02139-4307, U.S.A. E-mail: <leah@media.mit.edu>.

See <www.mitpressjournals.org/toc/leon/47/1> for supplemental files associated with this issue.

©2014 ISAST doi:10.1162/LEON_a_00470

of the disciplines that adopt open source rhetoric can lead to hollow, even meaningless, appropriations. Another point of controversy is the common assumption that "open source" is synonymous with "empowering" or "democratizing." It is not clear that this is the case, even in the realm of software.

There is a distinct difference between the socialist orientation of Richard Stallman's Free Software and the libertarian approach of Eric Raymond's Open Source Initiative.

In this article we focus on the translatability of "open source" to the field of architecture. Our inquiry revolves around three main questions: What is source code in architecture? What does or should "access to the source code" (and/or "accessibility") in architecture mean? How can an "open" architectural process be conceptualized based both on the ideas, practices and critiques of FLOSS and on longstanding discourses in architectural theory?

Open Source Architecture Today

The recent apparition of the term *open source architecture* in architectural discourse [10] suggests the possibility of combining advances in design and fabrication technologies with the ideas and practices of open source to reframe architectural design as a collective endeavor. The sphere of architectural practices that adopts the open source metaphor is non-homogenous; it contains discourses that range from a new kind of technological vernacular (e.g. "Architecture for Humanity" [11]) to hacktivism (e.g. "Hackitectura" [12]) and from a revisiting of old visionaries such as Christopher Alexander ("P2P Urbanism" [13]) to discussions of efficiency and mass customization (House_n [14]). While in some cases—such as the "1% Program" by Public Architecture [15] or the "Open Architecture Network" [16] by Architecture for Humanity—the model engages architects in collective authorship of architectural projects, the discourse on "open source architecture" also brings longstanding questions about the role of the user in the architectural design process to the surface.

The vision of a user-centered architecture, which has been revived by recent cyber-cultural discussions of democratization, has rich historical precedence. The employment of computation as a means to empower non-experts to design their own environments without the mediation of the architect was a highly popular vision in the 1960s and the 1970s. Designers and theorists, including Hungarian-born architect Yona Friedman, U.S. designer Christopher Alexander, Nicholas Negroponte and the Architecture Machine Group at the Massachusetts Institute of Technology (MIT), Dutch architect John Habraken, and many others [17], presented a number of proposals for participatory design systems that seem highly relevant today.

The neologism *open source architecture* is emerging as a broad and somewhat incoherent bricolage of these unfulfilled visions of technology-mediated participatory design with the pragmatic and ideological perspectives of FLOSS. The following definition by Carlo Ratti et al., recently published in *Domus* magazine, is an illustrative example:

> Open Source Architecture (OSArc) is an emerging paradigm describing new procedures for the design, construction and operation of buildings, infrastructure and spaces. Drawing from references as diverse as open-source culture, avant-garde architectural theory, science fiction, language theory, and others, it describes an inclusive approach to spatial design, a collaborative use of design software and the transparent operation throughout the course of a building and city's life cycle [18].

This quote positions open source architecture at the intersection of the historically rich discourse on user-driven design and the growing open source culture. The ambiguity and breadth of this definition are reminiscent of Stallman's criticism about the equation of "open source" with "participatory" [19]. This invites contemplation of whether open source architecture is merely an empty label or whether it characterizes a coherent and distinct effort to bring about and make sense of paradigmatic changes to the discipline of architecture.

Source Code and Access in Software

In software, source code can be described as a "fully executable description of a software system" [20]. It is a set of instructions that can be executed into a software application. The FLOSS definitions revolve around the idea that the source code is open to the public domain so that users can freely "run, copy, distribute, study, change and improve the software" [21]. The principle of accessibility, which ensures the meaningful character of these essential freedoms, is contingent on specific design requirements with which a program must comply in order to be characterized as "free" or "open source":

> The principle of transferring control to the user, however, does not only rely on the act of giving them access to the source code, but implies that the code itself is actually accessible (i.e. legible and not obfuscated) [22].

In the FLOSS community, the notion of accessibility refers to the direct link between the source code, in a legible and editable format, and the outcome of its execution. This ensures that access to the source code offers full control of the product (software) and allows for its study and modification. These principles are legally enforced through licenses (e.g. the GNU General Public License [23]), which are embedded in FLOSS source code and require individuals who use and modify code to share their new contributions.

In FLOSS, although the transmitter (programmer) and the source code are meticulously discussed in relation to the notion of accessibility, there is little reference to the user. The implicit expectation is that the user is an expert programmer, capable of studying, understanding and modifying source code. This points to a tension between the notion of accessibility as it is currently used by the FLOSS community and a more popular notion of accessibility as user empowerment—accessibility as the enabling of users with multiple levels of expertise to re-author products according to their needs and desires.

Source Code in Architecture

If providing "access to the source code" [24] in "the preferred form of the work for making modifications to it" [25] is one of the fundamental principles of FLOSS ideas and practices, then it is important to explore what source code means in the context of architectural design and what access to source code in a form suitable for making modifications entails.

A literal application of FLOSS practices to architecture would define open source architecture as an open sharing of the digital files that encode information on built artifacts. It can be argued that the source code → compiler → software product workflow finds its architectural analog in the procession from building information (drawings, models) to the mediator (contractor, builder) and to the final outcome (building).

In one vision of architecture's future, this analogy acquires accelerating force and relevance as digitization enables building processes that are increasingly specified by software. Building Information Modeling (BIM), which is currently gaining ground in architectural practice, is designed to concentrate and manage all information required to construct a building in one parametric and hierarchical digital model. The transition from traditional architectural drawings to a virtual representation of the building is assumed to remove a large part of the ambiguity of the transition from source code (building representation) to end product (building). In this scenario, contractors and builders are assumed to be mere executors of the instructions encoded in the BIM. The abundance of information in a BIM, from assembly instructions to lifecycle management data, makes the vision of shareable building information and the streamlining between design and construction appear more realizable.

In the ultimate realization of the literal application of the software workflow to architecture, machines replace contractors and builders. Large-scale digital fabrication technologies (e.g. building-scale 3D printers) enable an unambiguous translation of the digital description of a building to the artifact itself. The fabrication machines take on the role of compiler, exactly translating digital architectural designs into buildings. This brings to mind Clay Shirky's characterization of the physical aspects of construction as "simple executional steps at the end of a design manipulation process" [26].

However, these literal interpretations are vulnerable on several fronts. In practice, every step of the construction process—the translation of drawings and models into buildings—is vested with ambiguity and involves human interpretation, which is sensitive to physical context, personal skill and countless other variables. In his essay "Mapping the Unmappable" [27], Stan Allen likens builders to musicians: The score (drawing) offers instructions on how a piece will be performed but cannot determine the outcome, which is always dependent on the players.

Moreover, as long as humans are realizing designs, it is questionable whether additional information, such as information provided by BIMs, increases the predictable constructability of designs. Field studies in professional practice [28] demonstrate that the perceived complexity of a design stems from the translation of design information to construction information, which in turn is contingent not on the quantity of information but on its interpretation.

Finally, the impulse to objectify and disambiguate architecture by re-defining it as pure information processing is highly controversial. Although ambiguity is a challenge for the sharing and repurposing of an architectural design by different actors, it is commonly acknowledged as a valuable source of novelty, intuition and creativity [29]. Alternative computational design theories, such as shape grammars, challenge the equation of computation with rationalization and explicitness and assert design as a dynamic and improvisational process [30]. These approaches reject the idea that there is an isolatable, "objective" component in an architectural design that can be separated from the process of making. Interpretation and ambiguity are celebrated as the elements that allow different users to see different things in the same design representations, therefore leading to an explosion of creative repurposing. This attitude actively subverts the FLOSS principle of a one-to-one translation between code and product and welcomes the creative potential of the infinite number of these translations, according to the personal assumptions and ways of seeing of the different users/designers.

The inherent ambiguity in the processing of design information challenges the assumption that the sharing of design files (source) is enough to provide users with access to the architectural designs themselves (product). In contrast to software design, where there is a direct transition of code to product, architectural processes involve numerous levels of interpretation between a representation of a design and its realization. While the open sharing of architectural design files is possible [31] and may support new modes of collaborative design in architecture, it is unclear whether it supports democratization. Arguably, more meaningful accessibility is better supported through paradigms of openness that have historical precedents in architecture and could be fruitfully integrated with more literal interpretations of "open source."

Access in Architecture

This leads to our second question: What does and should accessibility mean in (open source) architectural design and how does this relate to the level of expertise of the user-designers involved? There is a distinctive difference between the notion of empowerment in the worlds of design and open source software. In FLOSS rhetoric, the empowered user is an elite programmer, while in discourses of design democratization the empowered user is a non-expert. Arguably, in order to fully democratize and "open" architecture, it is important to devise ways to engage the larger public in the processes of design.

> *The absence of source code in disciplines that adopt open source rhetoric can lead to hollow, even meaningless, appropriations.*

In software, the language of the design medium and the language of the end product are the same—programmers use software to generate software. In the case of architecture, complexity increases. To create a building, one must both be familiar with the medium in which architectural designs are encoded (e.g. digital 3D models, BIMs) and have the expertise and resources that are required in order to interpret this information. Furthermore, this entire process has to comply with the constraints imposed by building codes.

The vision of empowering users/inhabitants to operate within these complexities has played a central role in computational design research. In the early computational era (1965–1975), computer aids to design were primarily introduced for their ability to encode all necessary design constraints and ensure the production of adequate solutions. In his 1970 book *The Architecture Machine* [32], Negroponte framed this condition as initiating a "new humanism" enabled by machines, reconciling local desires with global constraints. The discourse of enabling the user/inhabitant to become a user/architect producing customized and responsive designs was soon extrapolated to the vision of complete removal of the professional architect from the process of design. This transition produced the "Design Amplifier" prototypes discussed in Negroponte's 1975 *Soft Architecture Machines* [33]. These personal "design partners" present remarkable affinities with the current research of MIT House_n's Open Source Building Alliance (OSBA), which proposes "intelligent" design engines enabling users to produce their own design configurations within a framework of design constraints encoded in their computational structures [34].

The concept of a platform, physical or computational, that allows for intuitive local solutions within a global framework of constraints constitutes a persistent paradigm of computationally mediated user empowerment in architecture. It can be traced from the pre-computational visions of the "megastructure"—a resilient structural framework in which users would have the ability to plug in ephemeral dwellings reflecting their ever-changing needs and desires [35]—to the early visions of computer-aided participatory design.

The ability to manifest constraints in interactive machines produced a series of design process models that, their creators postulated, would not only enable users to build buildings but would also help them become capable designers. In Negroponte's or Yona Friedman's proposals, the machine provides users with feedback allowing them to understand the implications of different design decisions for themselves and their community; thus users would gradually develop a level of design intelligence [36].

These examples do not exhaust the space of computer-aided participatory design but rather direct attention to a series of computational prototypes designed with the objective of empowering users to participate in design processes and collectively author the spaces they inhabit. Those involved in current initiatives such as House_n and other commercially available software for user-driven design are revisiting this approach to user-generated design [37].

However, in all these prototypes, implementations and visions, the user acquires access to design via a black box—a design software environment—that embodies its author's assumptions, knowledge and expertise. Users do not have the ability to access or modify this black box. Although pragmatically these platforms for participatory design can be argued to "open" design to non-expert users, they are themselves closed and inaccessible.

> *The impulse to objectify and disambiguate architecture by re-defining it as pure information processing is highly controversial.*

This tension suggests that we combine the ideas and practices of FLOSS with established frameworks of computer-aided participatory design to produce a hybrid structure that contains multiple sources and multiple layers of openness and accessibility. A fruitful approach might explore FLOSS's dictate that source code be provided in "the preferred form of the work for making modifications" [38], acknowledging that the preferred format for experts will be different from that for novices. Different formats—different levels of abstraction—provide accessibility to different kinds of users.

We therefore suggest to view the project of open sourcing architecture as inherently interdisciplinary, necessitating the collaboration of people with different skills and expertise. Taking the idea of open source architecture to its conceptual limits, one can imagine a system wherein groups and individuals have access to user-friendly design environments that give them control of the spaces they inhabit while also having access to the constraints and assumptions that underlie these environments. The questions of what the design characteristics of this platform/model should be and what technologies it would require can mobilize a rethinking of architecture's relation to the technology and social discourses of our time.

Conclusions

The phrase *open source* is being employed to rethink (and rebrand) the way that knowledge and artifacts are produced, distributed and used in architecture. Without dismissing the creative potential of misunderstandings, we feel it is valuable to look critically at the translational looseness that is often exhibited in the appropriation of open source in fields beyond software. In focusing on the recent emergence of the term *open source architecture,* we seek to frame the phrase as a set of practices that integrate established visions of user empowerment and democratization with the ideas and practices of the open source software movement.

The metaphoric and literal meanings of "access to the source code" in architecture expose a tension between the informational model of software development—wherein the transition from code to product is linear and predictable—and the inherent ambiguity in the interpretation of building information. Within this tension lie different definitions of accessibility, based either on efforts to eliminate ambiguity through the "objectification" of parts of the design process or on an alternative model that asserts this ambiguity as an inextricable part of the design process.

User empowerment is a necessary condition for meaningful accessibility. Looking at past and present computational platforms for user empowerment in architecture, we argue that their opacity and the constraints they encode run counter to open-source principles. As such, we recommend a model of layered openness in architectural code. This type of model for architectural code has the potential to help simplify and unify currently broad and amorphous definitions of open source architecture.

To conclude, the ideal "open architecture" requires more than openly publishing architectural designs; it demands a rethinking of the discipline's theory and practice—a re-diagramming of its processes and the roles of the subjects involved in them. A double inquiry into open source architecture, both from the perspective of a FLOSS scholar and an architectural historian, can expose the intricacies of the integration of ideas and philosophies from these disciplines and engender new frameworks for architectural design.

References and Notes

Unedited references as provided by the authors.

1. The European Commission adopted this acronym in 2002 as a replacement of the initial FOSS (F/OSS), which did not include the Spanish term *libre*.

2. D. Bradley, "The Divergent Anarcho-utopian Discourses of the Open Source Software Movement," *Canadian Journal of Communication*, Vol. 30, No. 4 (2005): 585–611.

3. The quote "The future is open source everything" can be found in multiple online sources as attributed to Linus Torvalds, the inventor of Linux; however its provenance remains unconfirmed.

4. Bradley [2] p. 588.

5. R. Stallman, "Why Open Source Misses the Point of Free Software," <www.gnu.org/philosophy/open-source-misses-the-point.html>.

6. Stallman [5].

7. Bradley [2] p. 587.

8. M. Tiemann, "Future of Cygnus Solutions: An Entrepreneur's Account," in C. DiBona, S. Ockman and M. Stone, eds., *Open Sources: Voices from the Open Source Revolution* (Sebastopol, CA: O'Reilly & Associates, 1999).

9. See, for example, C. Thompson, "Build it. Share it. Profit. Can Open Source Hardware Work?" (2008) <www.wired.com/print/techbiz/startups/magazine/16-11/ff_openmanufacturing>.

10. One of the first discussions of the concept of "open source architecture" was in D. Kaspori, "A Communism of Ideas: Towards an Architectural Open Source Practice," *Archis* **3** (2003): 13–17.

11. "Architecture for Humanity," <http://architectureforhumanity.org/>.

12. "Hackitectura.net | Arquitectos, Programadores y Artistas Proyectando en la Convergencia de Espacio Fisico y Digital," <http://hackitectura.net/blog/>.

13. N.A. Salingaros, *P2P Urbanism* (2010) <http://zeta.math.utsa.edu/~yxk833/P2PURBANISM.pdf>; "Peer to Peer Urbanism," <http://p2purbanism.blogspot.com> and <http://p2pfoundation.net/Peer-to-Peer_Urbanism>.

14. "MIT House_n," <http://architecture.mit.edu/house_n/>.

15. "The One Percent Pro Bono Design Program of Public Architecture," <www.theonepercent.org/>.

16. "Worldchanging | Evaluation + Tools + Best Practices," <http://openarchitecturenetwork.org/>.

17. A major event that brought together seminal figures who were at the time active in participatory design was the 1971 Conference on "Design Participation," organized by the Design Research Society in Manchester.

18. C. Ratti et al., "Open Source Architecture," *Domus*, Vol. 948 (June 2011) <www.domusweb.it/en/op-ed/open-source-architecture-osarc-/>. This article is also the basis for the current Wikipedia definition of "Opensource Architecture."

19. "What is free software?" <www.gnu.org/philosophy/free-sw.html>.

20. M. Harman, "Why Source Code Analysis and Manipulation Will Always Be Important." *10th IEEE International Working Conference on Source Code Analysis and Manipulation* (Timişoara, Romania, 12–13 September 2010).

21. See Ref. [19].

22. See Ref. [19].

23. "GNU General Public License," <www.gnu.org/copyleft/gpl.html>.

24. See Ref. [19].

25. See Ref. [23].

26. C. Shirky, "Generalizing peer production into the physical world" (2007) <http://finance.groups.yahoo.com/group/decentralization/message/6967>.

27. S. Allen, "Mapping the Unmappable: On Notation," in S. Allen and D. Agrest, *Practice: Architecture, Technique and Representation* (London: Routledge, 2006).

28. J.M. Lobel, *Building Information: Means and Methods of Communication in Design and Construction,* Smarchs Thesis (Cambridge, MA: MIT Department Of Architecture, 2008).

29. An extensive discussion on the importance of ambiguity in design can be found in G. Stiny, "New Ways to Look at Things," *Environment and Planning B: Planning and Design* (1998): 68–75 (Anniversary Issue).

30. See, for example, G. Stiny, *Shape: Talking about Seeing and Doing* (Cambridge, MA: MIT Press, 2006).

31. For example, the practices of architectural design file sharing have been extensively implemented by the Open Architecture Network. See "Find current projects | Worldchanging," <http://openarchitecturenetwork.org/projects/results>.

32. N. Negroponte, *The Architecture Machine* (Cambridge, MA: MIT Press, 1970).

33. N. Negroponte, *Soft Architecture Machines* (Cambridge, MA: MIT Press, 1970).

34. K. Larson et al., "Open Source Building: Reinventing Places of Living," *BT Technology Journal* **22**, No. 4 (2004).

35. An indicative example of the megastructure as a locus of design participation can be found in Yona Friedman's influential "Mobile Architecture" manifesto. See Y. Friedman, *L'Architecture Mobile* (Brussels: Centre d'études architecturales, 1968).

36. These "machines" are Negroponte's "Design Amplifier" and Friedman's "FLATWRITER," described in *Soft Architecture Machines* and in Friedman's *Toward a Scientific Architecture*. See Negroponte [33] and Y. Friedman (transl. Lang C.) *Toward a Scientific Architecture* (Cambridge, MA: MIT Press, 1975).

37. The two most common types of such tools are "configurators," which guide users through different options in order to configure a design (for example see Blu Homes, <www.bluhomes.com/>) and "design recommendation engines," whereby design solutions are matched to user profiles (for example, see the Home Genome Project at MIT, <http://cp.media.mit.edu/research/77-home-genome-project>).

38. See Ref. [23].

Manuscript received 5 March 2012.

Theodora Vardouli is an architect and researcher currently pursuing a PhD in Design and Computation at the MIT Department of Architecture. Through writings, projects and teaching, she traces relationships between design democratization and computation from the 1960s to the present.

Leah Buechley is an associate professor at the MIT Media Lab, where she directs the High-Low Tech research group. The High-Low Tech group explores the integration of high and low technology from cultural, material and practical perspectives, with the goal of engaging diverse groups of people in developing their own technologies.