An Open-Source Paradigm in the Responsive Architecture Studio

In the last decade, rich participatory open-source communities, and open-source hardware and software, have emerged with resulting opportunities for change in architectural education. We discuss a pedagogical approach for responsive architecture, involving open source content, first-hand design experiences , and teaching experiences in architectural studio.

SUSAN FROSTEN
Philadelphia University

KIHONG KU
Philadelphia University

JONATHAN GRINHAM
Harvard University

OPEN SOURCE AND ARCHITECTURE

Open source is a popularized term and concept associated with the notion of participatory culture. Originated in software programming, the aspects of democratizing production have been broadened to a variety of contexts such as open source culture, open source hardware, open content, and related concepts such as creative commons, and participatory culture. Open content creation including knowledge production such as Wikipedia, or artistic work by amateurs or professionals such as Flickr, YouTube, etc., biotechnology (e.g., BioBricks Foundation), electronics, design and education (Cheliotis 2009; Ceraso and Pruchnic 2011; Hope 2008; Voyce 2011) are just a few examples. Open source is also associated with leveraging voluntary labor in the form of crowdsourcing to outsource portions of a larger task to an indefinite group of volunteers, or 'prosumption' to involve consumers in the production of good and service such as usability, beta testing, and citizen journalism of nonprofessionals' contribution to covering news events, and participatory and social media (Ceraso and Pruchnic 2011).

Jenkins et al. (2006) discuss a 2005 study conducted by the Pew Internet and American Life project (Lenhardt & Madden, 2005). According to this study more than one-half of all American teens—and 57 percent of teens who use the Internet—could be considered media creators who created a blog or webpage, posted original artwork, photography, stories or videos online or remixed online content into their own new creations. One-third of teens share what they create online with others, 22 percent have their own websites, 19 percent have blogs, and 19 percent remix online content (Jenkins et al. 2006). A major shift toward the decentralization of knowledge changes both the perception of who produces content and who possess authoritative view on content.

Aspects of the open source paradigm and participatory culture have drawn parallel speculations in architecture. Ratti et al. (2011) propose open source

architecture (OSArc) as an emerging paradigm which implies an inclusive process through collaborative use of design software and transparent operation throughout the life cycle of design, construction, and operation of buildings, infrastructures and spaces. Accordingly OSArc includes (1) new funding models of crowd-funding strategies (e.g., Kickstarter, Sponsume), offering mass ownership appealing to squatters, refugees; (2) inclusive forms of engagement through crowd-funding seen as a spatial version of Hacktivism, which may suffer drawbacks such as project bifurcation, abandonment, incompatibility with existing buildings ; (3) common, open, modular standards for hardware compatibility, networks of knowledge, ideas, and remote collaboration; (4) design for mass customization through parametric tools, open source codes and scripts, BIM, rapid prototyping and 3D printing technologies involving laypeople as decision-making agents, sharing information, and optimizing production; (5) construction with open source hardware for kinetic or smart environments integrating software, hardware, mechanisms, with embedded sensing and computing within the 'internet of things'; and (6) networked real-time monitoring and system feedback for personalization during maintenance and operation. Taking into account user- and environmentally-responsive mechanisms, Haque (2002) argues that OSArc is a participatory spatial operating system that allows participants to design and reappropriate their own spaces and share design problems to build social space.

RESPONSIVE ARCHITECTURE

As open source architecture suggests, responsive architecture is an interaction systems within which users create their own programs. Advancements in robotics technology combined with the demand for smart, sustainable and user friendly environments have drawn attention towards kinetic and responsive architecture. A number of precedents illustrate embedded intelligence and new tectonic opportunities that offer spatial qualities of adaptable physical environments supported by digital technologies.

Several architectural researchers and designers have examined the aesthetics of such architectural kinetics and proposed taxonomies and design methodologies for this emerging paradigm. Moloney (2011) offers a framework for architectural facades based on analytical diagrams and time-lapse images of dynamic and animated surfaces. This kinetic paradigm suggests a shift from designing a final static product to an adaptable and interactive process. Thus the traditional role of architects as spatial consultants requires adopting complex subassemblies of dynamic systems which integrate product design, computer science, engineering, mechanical and electrical engineering, behavior sciences and material sciences, and other fields.

The multidisciplinary aspects of this emerging domain are easily recognized by the variety of relevant terms used in research and practice: ubiquitous computing, pervasive computing, interactive architecture, responsive architecture, kinetic architecture (Zuk 1970), transformable design (http://www.hoberman.com/) and architectural robotics (Green and Gross 2012). Fox and Kemp (2009) survey the landscape of interactive architecture approaches driven by practical needs such as environmental sustainability, aging, changing lifestyle patterns, new sensual experiences and implications. These developments build on the advancements in building technology and embedded technology (e.g., microprocessors, sensors, actuators, etc.) which acknowledge the reality and demand for pervasive computing (McCullough 2002). New typologies in architecture offer possibilities for new information spaces that integrate both the physical and virtual spaces (Ku and Grinham 2013).

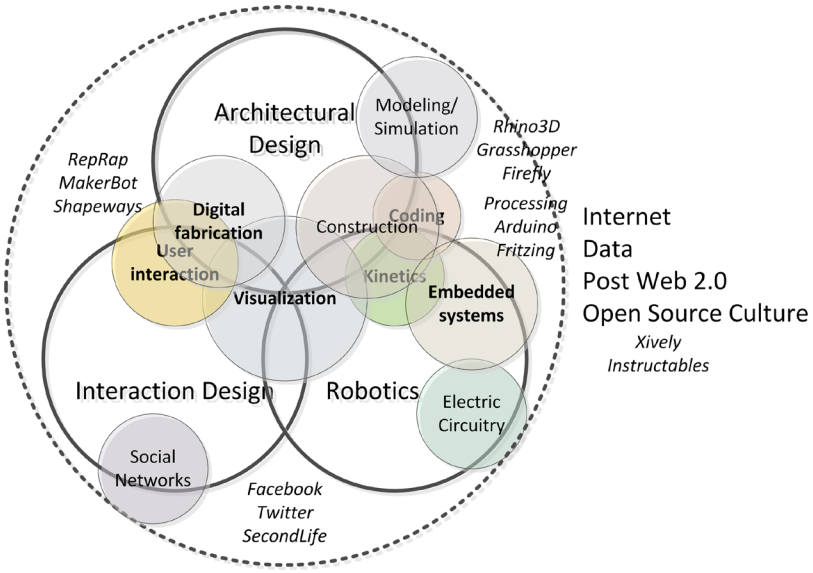


Figure 1: Core domains of responsive architecture.

Figure 1 shows the key domain knowledge areas that contribute to this design process. Architectural design, robotics, and interaction design, contribute to this interdisciplinary field of responsive architecture. It is necessary to understand how the complex relationships of algorithmic logic, simulation, physical prototyping, robotics implementations, etc. impact tectonic possibilities, what design tools are adequate, and how knowledge boundaries become increasingly assimilated into the architectural domain. It became obvious through our own design explorations that open source culture and the internet is an important driver pushing technology, design, and dissemination of responsive architecture forward.

OPEN SOURCE SOFTWARE

In 2001 Ben Fry and Casey Reas released Processing, a programming language and development environment for the visual arts. Fry and Reas developed the programming software while at MIT's Media Lab under the direction of John Maeda. The programming software uses the graphic capacity of Java programming with simplified and new features geared toward students, artists and design professionals. The importance of Processing is not necessarily that it is an easy to use and powerful programming language developed specifically for the visual arts (it is), but rather, that it is an open source software platform. Processing itself is the result of other open source software that provided guidance and components for Processing (Reas & Fry 2007). When it was released, Processing also unveiled the framework for a collaborative community. When the software launched, it brought with it a network geared toward creating a community of connected users and producers. This online community included tutorials, code banks, forums and 'wiki' pages where like-minded programmers of all skill levels could come together and share ideas, codes and continue to develop the Processing environment. Fry and Reas describe the Processing website as a 'communication hub' and describe other Processing-based websites' willingness to share source codes. Jared Tarbell of Complexification.net states, "opening one's code is a beneficial practice for both the programmer and the community. I appreciate modifications and extensions of these algorithms." (Reas & Fry, 2007). This willingness to share provides rich soil for learning. Designers seeking to learn Processing are not required to create immediately, nor do they need to seek out a centralized learning point. Rather, they can begin were others have left off, learn from their peers and build upon them. Open source software like Processing

provides the necessary framework for a participatory culture. This framework is described by Elinor Ostrom's 'Mechanisms of Joint Governance of Commonly Accessible Resources.' Simply stated, by providing a free good that allows users themselves to access the source code, open source software produces a low hurdle for participation. Users are motivated to use the free code and make changes where the code is weak or to develop functions further. Because it is free, subsequent versions or refinement of code are markedly unsustainable for private sale (Shirky, 2010). Nowhere is this better explained then through the development of Arduino.

OPEN SOURCE HARDWARE

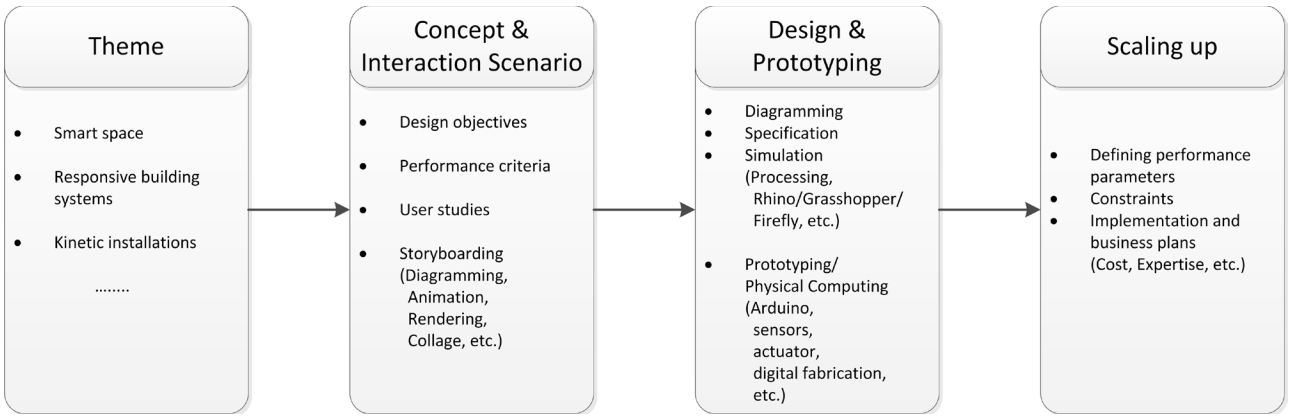
Processing, through an open source licensing, gave birth to two new platforms: Wiring in 2003 and Arduino in 2005. Both platforms were designed for artists to learn how to program microcontrollers—small computers with a single, integrated circuit containing a processor core, memory and programmable input/output peripherals. These platforms are an example of the means by which we see a re-emergence of responsive architecture (We have chosen to focus on Arduino due to its popularity in academic communities including the Architecture Association in London, UCLA's undergraduates in the Design | Media Arts program and New York University's graduate ITP program—all of which have produced valuable online communities that have contributed to this study.) Massimo Banzi and David Cuartielles founded Arduino in 2005 seeking to develop an open source, inexpensive prototyping system based on the processing language. Much like Processing, Arduino launched with an online community where users can contribute to, and borrow from, their peers. Arduino provided a platform for development that paired Do-It-Your-Selfers (DIY) with academics and professionals around the world. The Arduino integrated development environment's (IDE) ease of use, popularity, and access to the Atmel AVR microprocessor has also birthed more than a half-dozen Arduino-based microcontrollers, including, Lillypad, Microduino, Fabduino, and Printo, to name a few.

These networks also provide a secondary aspect of open source software, known as 'off-the-shelf' hardware. The use of 'off-the-shelf' hardware and circuits offers a similar framework to the use of open source software. In this case, the hardware is front-loaded with information. The use of 'off-the-shelf' hardware ensures consistency of products, specifications and results. More importantly, like Processing, Wiring and Arduino, the suppliers of 'off-the-shelf' hardware have produced information-rich online participatory cultures. Vendors such as, Sparkfun.com, Adafruit.com, Robotshop.com and Allelectronics.com provide their consumers with outlets to become information producers. Product pages provide areas for product reviews and forums where consumers can find valuable codes, circuits diagram and links to experiments using the specific products. Sites like Sparkfun.com also include tutorials by staff members and customers. Programs such as Fritzing, an open source platform, allow users to design circuits through computer software that includes libraries of 'off the shelf' hardware. Users can import parts, design and share their project through Fritzing.

A PEDAGOGICAL FRAMEWORK

Acknowledging the potential of open source architecture in responsive architecture, our study explored the following pedagogical questions:

- How is architectural education changing due to these innovations?
- What are the related research skills and information literacy requirements for students?



2

- What are the opportunities and challenges for architectural education?

To answer these questions, we investigated design research along three prongs:

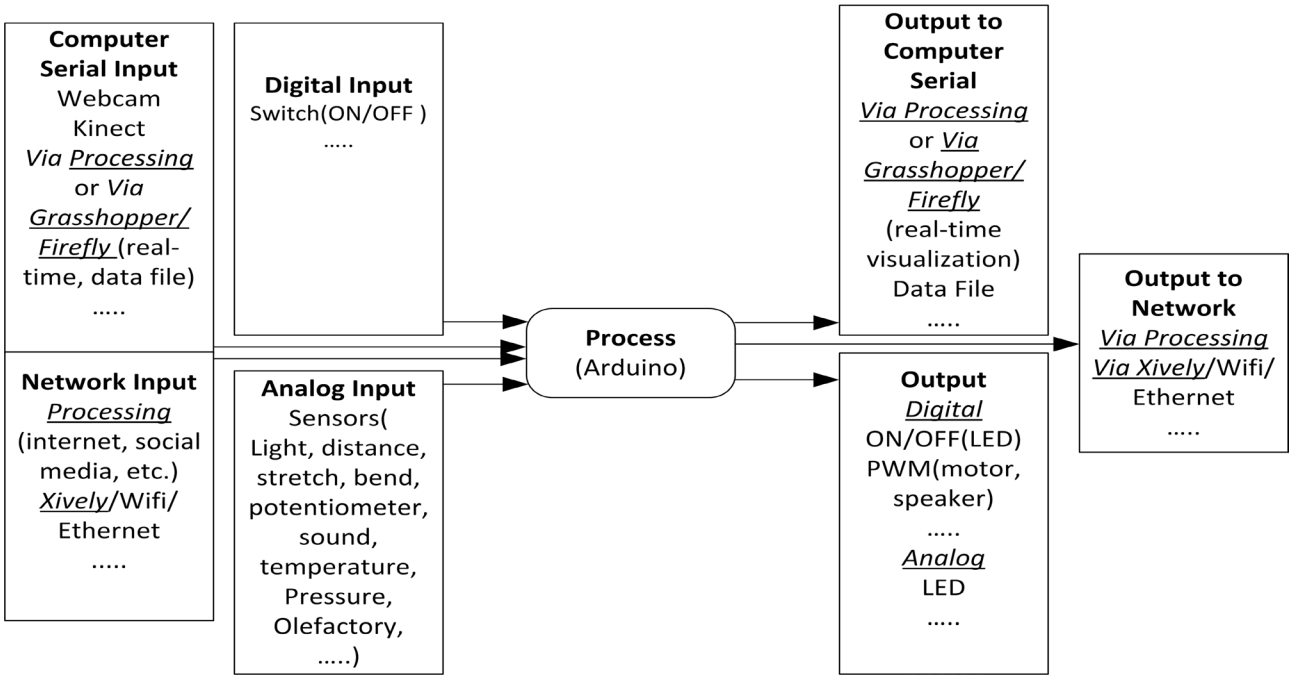
- Review and mapping of open-source content of digital design
- First-hand experiments with open-source hardware and software by the authors on responsive architecture design projects
- Responsive design studio course development

Our explorations in responsive architecture are based on a framework involving systematic steps of defining a theme, defining design goals and developing interactive scenarios, designing and prototyping, and finally specifying requirements for full scale implementations (Figure 2).

Thematic explorations have involved kinetic installations which facilitate interaction between users or interaction with users, responsive structures and building skins that respond to environmental conditions such as lighting, sound, temperature, etc., and smart spatial interfaces that bridge physical and virtual spaces and offer connectivity between users or sensors and actuators. Within these themes, design goals and user experience and interaction scenarios are developed to establish user needs and system requirements. This stage involves defining and envisioning scenarios through user studies, story boarding, renderings, animations, and collages. Based on the outcomes, design and specifications are developed and these will be used to evaluate the success of the project outcomes during prototyping studies and full scale implementations. Simulations through the use of renderings, virtual prototyping via software (e.g., Processing, Rhino 3D, Grasshopper, etc.) are utilized to study design options. The prototyping phase involves development of interactive mock-ups that utilize open source hardware and software and techniques of rapid prototyping and digital fabrication. The final phase involves scaling up the prototype and may end at considering plans and implications of such developments because of practical and funding constraints, with the intent to allow future development if interested parties can be attracted and involved. Thus potential outputs may establish performance parameters, constraints, specifications and plans for future implementations, or result in full scale implementations.

As we see in the design process diagram, the development process of responsive architecture demands multidisciplinary understanding and foundational skills in architectural design, interaction design, and physical computing and simulation. Developing the required multidisciplinary knowledge in these areas is one of the challenges in the architectural design curriculum.

Figure 2: Framework of responsive architecture design process.



3

Figure 3: Input-Process-Output interface diagram based on the Arduino platform.

OPEN SOURCE PLATFORMS FOR DESIGN AND PROTOTYPING

For prototyping purposes, one of the open source hardware platforms chosen by the authors and used for design and teaching is the Arduino microcontroller. In parallel, Processing is the open source software interface utilized to directly control or interface with the Arduino. Both platforms are compatible with each other and offer convenient connectivity via serial connections (USB cable) through the computer. In addition to Processing, Firefly—a Rhino/Grasshopper visual programming interface provides connectivity to Arduino. The Arduino offers a flexible platform to extend real-time data or stored data from inputs such as various sensors, or data streams channeled through the computer (e.g., camera, Kinect) or data from the internet. The Arduino microcontroller processes the data based on algorithms that are custom programmed by the user to generate various forms of actuations such as real-time visualizations via Processing, or physical actuators (LED, motor, speaker, etc.) or via the internet to other computers or microcontrollers (Figure 3).

OPEN SOURCE COMMUNITIES RELEVANT TO COMPUTATIONAL DESIGN

The incredible growth of online communities within the architecture and design practices, evidence the broad user base of various software and hardware tools some of which are commercial tools and other open source tools. We conducted a review of a few selected websites (Table 1). The review included online communities for open information and knowledge sharing, personal blogs, electronics suppliers and professional service providers. Within the category of online communities, based on the user types, four categories were observed: (1) developer-user communities which were initiated by developers and then opened to broad user bases; (2) user communities initiated by users to share ideas, information among users; (3) commercial provider initiated communities which combine user communities with commercial publication channels; and (4) academic classroom spaces which share classroom content with the larger online community. These communities share knowledge, information, through tutorials or user forums and downloadable code (e.g. processing, Arduino, etc.), 3D geometric specific families (e.g., RevitCity), and others share also job postings and relevant news articles.

Category	Name	Components	Participant Type
Online communities	Processing.org	Download Examples Tutorials References Forum/Support Shop/Buy	Open source software developer-user community
	Openprocessing.org	Examples Course examples Collections Shop/Buy	Open source design share-user community
	Arduino.org	Download Shop/Buy Tutorial Examples References Support/Forum Blog	Open source hardware developer-user community
	RevitCity.com	Forum Downloads Gallery New/Articles Resources Jobs FAQ	Commercial software user community
	Designreform.net Designbymany.com Case Consulting	Publication/tutorials Community Consulting	Commercial training provider on various tools
	Instructables.com	Explore Create Contests Forums	DIY user community
	http://www.wikispaces.com/	Online classrooms	Academic classrooms
Blog	http://shiffman.net/	Books Teaching Blog Download	Educational provider on open source tools
	http://www.jeremyblum.com/	Blog Tutorials/Books Portfolio examples	Educational provider on various software/hardware tools
	http://www.plethora-project.com/	Video tutorials Portfolio examples Blog Code library	Educational provider on various design tools
Training provider	http://lab.modecollective.nu/	Online tutorials Workshop arrangements	Commercial training provider on various tools
	http://www.fabfoundation.org/fab-labs/what-is-a-fab-lab/	Technical prototyping platform Knowledge sharing network	Digital fabrication and computation platform
Material vendors	https://www.sparkfun.com/	Products Blog Tutorials Videos Classes Support	Commercial electronics supplier with tutorials and user discussion board and support
	http://www.adafruit.com/	Shop Blog Learn Forum	Commercial electronics supplier with tutorials and user discussion board and support

4

Some of the key features of vibrant communities include interactive content, membership rankings based on user contribution. For example, RevitCity's use of a membership-based community highlights an important motivation for user generated content within online communities. Social motivations reinforce the personal motivations and new communications networks encourage membership and sharing, providing support for autonomy and competence (Shirky 2010). The use of membership provides a rich intrinsic motivation for sharing, and the membership approach to problem solving produces a shifting from, 'I did it' to 'we did it', resulting in direct social feedback and a sense of connectedness. In

Figure 4: Various categories of open source communities and websites.

RevitCity's case, membership ranking also serves as a way of substantiating user generated content. Membership ranking is a peer-reviewed system that not only produces trust within a community, but also provides extrinsic motivation to share, thereby increasing user content. This increase is evidenced by RevitCity's growth (based on interview conducted with Hiroshi Jacobs, founder of RevitCity.com, on January 6th 2011).

Other online sources such as blogs and websites maintained by commercial training providers offer information of personal research and structured book content. In the case of commercial providers, free content previews are offered to attract potential customers for additional offerings.

From an information consumer standpoint, these online sources can be brought into the design and prototyping process in a number of ways: (1) tutorials, references, guides, are useful to study the basics of the software interfaces and fundamentals of code or electronics to get started; (2) design-share communities such as OpenProcessing.org or DIY communities such as Instructables are helpful to inspire users of the creative potential of the tools; (3) various forums are generally useful for troubleshooting issues with code/algorithms and circuitry, although when confronted with novel problems, the drawback is that specific problems may not be resolved because of the lack of expertise in the user base or difficulty of finding the expert (Ku and Grinham 2013); (4) code libraries are often integrated and accessible from within the programming interface of the Arduino and Processing but also available for download. These are generally helpful to simplify coding processes such as the Servo library example described earlier which simplifies the user coding of pulse width modulation to turn servo motors; (5) material supplies offer shopping guides of hardware and link user feedback; (6) open creative examples can also be customized by other users to building upon other's work.

OPEN SOURCE IN THE DESIGN STUDIO

In the context of the authors' own design explorations and teaching of responsive design studios, open source has proved to be helpful in various aspects. As we discussed earlier, the design and prototyping of responsive architecture comprises at the core architectural design, interaction design, and physical computing. In the authors' own early design explorations, the focus was on learning and exploring relevant code and hardware applications for prototyping purposes. Thus open source resources were primarily referenced for learning and applying library objects. Subsequently, the lessons learned from these coding and hardware prototyping experiences became the source for future studio teaching and information sharing.

Our experience of teaching the responsive architecture studio to fifth year undergraduate architecture students revealed the needs and benefits for expanding open source practices. Particularly architecture students who are fairly well equipped with foundations in architectural design, with some understanding of interaction design, and some experience in parametric modeling and algorithmic design commonly wish for more learning in coding. Some of the difficulties are in understanding algorithmic logic; others are indicating the difficulty of composing more complex sequences that involve classes, simplification through looping, and sub-procedures. The benefit of utilizing open source code is that sometimes more complex code examples already exist and can be customized to fit the specific objectives of custom scenarios. But one of the challenges in such cases is

	Finding & Learning	Using & Customizing	Sharing
Coding instructions	**To be able to use and develop basic and custom algorithms for interactivity**	**To be able to find, understand, use and customize applied code examples**	Coding examples properly formatted and annotated for others to understand and use
Coding libraries	**To be able to find and understand libraries that can simplify operations or enable new functions**	Utilizing libraries to achieve user specific program objectives. Customizing code libraries requires somewhat advanced skills and knowledge (e.g., resolve conflicts between libraries). Regular user would not need to have this ability	Share custom libraries that can help others to achieve new functions or simplify functions. Regular user would not need to have this ability
Kinetic mechanisms	To be able to develop kinetic mechanisms *Analyze kinetic behavior based on mechanisms*	*To be able to apply kinetic typologies or mechanisms*	Instructions of kinetic construct development process
Hardware	**To be able to identify, specify, acquire and use off-the-shelf hardware**	Customizing/hacking hardware	Instructions for sensors, actuators, network applications Circuit diagrams
Virtual prototyping/ Visualization software	**To be able to use Grasshopper/Firefly or Processing for virtual prototyping of interaction scheme**	To understand how to interchange coding schemes between different software environments	**Build 3D component library of models for prototype assembly diagrams**
Digital fabrication	*To be able to use adequate fabrication tools (e.g., laser cutter, 3D printer, etc.)*		*Document fabrication methods and design documentation*
Open design schemes	*Identify and analyze precedents of interaction schemes, kinetic constructs, and smart spaces.*	*To be able to apply principles of interactivity, kinetic constructs and smart systems to specific design problems, or customize existing interface or systems to new problem contexts.*	*Develop interfaces instructions or plug-and-play schemes that allow code modifications or custom interaction by users* *Develop assembly systems that can be replicated by users (DIY kit instructions* *Document, video record working proof-of-concept prototypes*

5

that it is not always easy to find similar code examples that can be appropriated. Table 2 maps key activities involving open source content and skills. The activities in bold fonts indicate areas that facilitate skill building and help to improve productivity and are usually more emphasized from the learning end. The areas in italic fonts refer to skills that architectural students are generally prepared with and proficient in and being able to share with others. Students are able to design open systems of kinetic architectural interfaces that allow user input and customization of the interaction schemes by future users. An interesting observation is that students manage well to explore opportunities for kinetic installations but may not necessarily connect the concepts of responsive interfaces with smart system concepts.

DESIGNING AN OPEN DESIGN COMMUNITY PLATFORM

While we observed a large number of open source websites, the authors utilized the University Blackboard courseware and University server for information, data, document, code, and file sharing. The Blackboard site offered user

Figure 5: Open source activities in the responsive architecture design studio.

content sharing via discussion boards for reading discussions, blogging for seminar and research topic sharing. In parallel, a University server was used to post and share presentations of ongoing design work, codes, digital models, and all relevant design content. In addition, the students posted design progress work in the format of videos and Prezi presentations on the Adobe Behance (http://www.behance.net/) website (examples from the studio taught in spring 2014 can be searched with the tag 'arch508spring14). For content sharing and posting, the authors may explore other open platforms such as Wikispaces in the future.

CONCLUSION

Within the responsive architectural studio the impact of open source is obvious and interwoven with design teaching and learning. The architecture students exhibit strength on the information sharing end of developing open design schemes, kinetic constructs and architectural interfaces. While in the areas of coding and physical computing, some projects achieve interesting software and hardware schemes; in general the architecture students indicate that they wish to gain more in-depth knowledge of coding and electronics to feel more confident. With the growing influence of open source in the design computing curriculum, it is important to consider how to properly use and customize open source information, and understand various licensing agreements such as Creative Commons. Accordingly, it also needs to be established how to document, format and annotate creative products for open sharing. At this stage our explorations on open source have focused on fostering internal communities of practice within the University's design studio and we are looking for opportunities across colleges and universities, and broadening the collaboration to open source communities.

ACKNOWLEDGEMENTS

This work was supported by a Nexus Grant from the Center for Teaching Innovation & Nexus Learning (CTinL) at Philadelphia University. Its contents are solely the responsibility of the authors and do not necessarily represent the official views of CTinL.

ENDNOTES

Cheliotis, G. (2009) From open source to open content: Organization, licensing and decision processes in open cultural production, Decision Support Systems, 47, pp. 229–244, Elsevier.

Ceraso, A., and Pruchnic, J. (2011) Open source culture and asthetics, Criticism, Summer 2011, Vol. 53, No. 3, pp. 407–438. Wayne State University Press, Detroit, Michigan 48201-1309.

Fox, M. and Miles, K. (2009). Interactive Architecture. New York: Princton Architectural Press.

Green, K. and Gross, M. (2012) Architectural robotics inevitably, Interactions, Vol. XIX.1, January February 2012, Association for Computing Machinery.

Ku, K. and Grinham, J. (2013) 4D Environments and Design: Prototyping Interactive Architecture, ARCC News and Reports, http://arccweb.org/newsletter/category/newsletters/37-1-spring-2013/ (accessed March 10, 2014).

Haque, U. (2002) Hardspace, softspace and the possibilities of open source architecture, http://www.haque.co.uk/papers/hardsp-softsp-open-so-arch.PDF (accessed March 10, 2014).

Hope, J. (2008) Open Source Revolution in Biotechnology, Cambridge, MA: Harvard University Press.

Jenkins, H., Purushotma, R., Clinton, K., Weigel, M. and Robison, A.J. (2009) Confronting the Challenges of Participatory Culture: Media Education for the 21st Century. Digital Media and Learning. The MacArthur Foundation. http://mitpress.mit.edu/sites/default/files/titles/free_download/9780262513623_Confronting_the_Challenges.pdf (accessed March 10, 2014).

Lenhardt, A., and Madden, M. (2005) Teen Content Creators and Consumers, Pew Internet & American Life Project, Washington, DC.

McCullough, M. (2004) Digital Ground: Architecture, Pervasive Computing, and Environmental Knowing, MIT Press.

Moloney, J. (2011) Designing kinetics for Architectural Facades: State change, Routledge.

Ratti, C., Antonelli, P., Bly, A., Dietrich, L., Grima, J., Hill, D., Habraken, J., Haw, A., Maeda, J., Negroponte, N., Obrist, H.U., Reas, C., Santambrogio, M., Shepard, M., Somajnl, C., and Sterling, B. (2011) Open source architecture, op-ed, Domus 948.

Reas, C., & Fry, B. (2007). Processing: A Programing Handbook for Visual Designers and Artists. Cambridge, MA: MIT Press.

Shirky, C. Cognitive Surplus. New York: Penguin, 2010.

Voyce, S. (2011) Toward an open source poetics: Appropriation, collaboration, and the commons, Criticism, Summer 2011, Vol. 53, No. 3, pp. 337–375. Wayne State University Press, Detroit, Michigan 48201-1309.

Zuk, W. (1970) Kinetic Architecture, Reinhold.