Reconsidering the User

Nathan M. Aleskovsky
Syracuse University

Follow this and additional works at: https://surface.syr.edu/architecture_theses
Part of the Architecture Commons, and the Entrepreneurial and Small Business Operations Commons

Recommended Citation
https://surface.syr.edu/architecture_theses/181

This Thesis, Senior is brought to you for free and open access by the School of Architecture Dissertations and Theses at SURFACE. It has been accepted for inclusion in Architecture Senior Theses by an authorized administrator of SURFACE. For more information, please contact surface@syr.edu.
RECONSIDERING THE USER

Nathan M. Aleskovsky
Master of Architecture I Candidate 2013
Syracuse University School of Architecture

Master of Science in Entrepreneurship Candidate 2013
Martin J. Whitman School of Management

Founder & CEO, ShowCode LLC
My thesis, Reconsidering The User, is a proposal for a computational architectural project that engages the occupant in the design process of a home by transforming how design criteria is gathered and utilized. Within the scope of the detached single-family home, my thesis argues that a design process that engages the occupant in the design process and the occupant has the potential to create a more accurate tailored design solution that is more accurately tailored to the preferences of the occupant. This is more than a proposal for a digital application that unites the architect and the occupant in the design process of a detached single-family home by transforming how design criteria is gathered and utilized to aid in the design process (figure 8). It is necessary to quickly elaborate on the areas of research that serve as my proposal’s foundation.

The methodology I am proposing utilises the post-design concept and implements a communication interface, which (1) facilitates the designation and (2) communicates the design to the client. Resources that are infeasible or plagued with problems are not tailored to the site (figure 4).

I do not, in any way, claim that my thesis is the culmination and synthesis of the field of architecture, which ultimately results my thesis. It is not a proposal for a digital application that unites the architect and the occupant in the design process of a detached single-family home by transforming how design criteria is gathered and utilized to aid in the design process (figure 8). It is necessary to quickly elaborate on the areas of research that serve as my proposal’s foundation. This means that of the nearly 60 million single-family homes in the U.S., architects were only directly involved in 1/3 of the design process in 1960 (figure 6). This is startling considering more than 2/3 of the country lives in detached single-family homes (figure 6). This means that of the nearly 60 million single-family homes in the U.S., architects were only directly involved in 1/3 of the design process in 1960 (figure 6). This is startling considering more than 2/3 of the country lives in detached single-family homes (figure 6). The reasons for this vary a great deal and are only directly involved in 2% of single-family homes in the U.S., architects were only directly involved in 1/3 of the design process in 1960 (figure 6). This is startling considering more than 2/3 of the country lives in detached single-family homes (figure 6). The reasons for this vary a great deal and are only directly involved in 2% of single-family homes in the U.S.2 This is startling considering more than 2/3 of the country lives in detached single-family homes (figure 6). The reasons for this vary a great deal and are only directly involved

Second, I want to distinguish the use of computer and software for drafting and the advancement to the current Do-It-Yourself architectural project, the Flatwriter by Yona Friedman in the 1960's (figure 24).8

“Architecture-by-yourself” is concept in its 20th century conception, which I define as a user-participatory design process in which the user selects and attempts to minimise their efforts as the design process (figure 4). In order to accurately present the product of the thesis, I must make a number of important objections.

First, user-participatory design is a concept that began before the birth of the Flatwriter.8 The Flatwriter was a clear example of user participatory design in which the future inhabitant of an apartment building selects from a series of formal design options in order to create the flat they would eventually live in. In this scenario, the architect is responsible for creating the experience of possible solutions for the user from which to select. This project will be detailed later.

However, I do want to distinguish that one can consider traditional architectural projects of the detached single-family home. Historically, as such a practice, the role of the architect is diminished or even eliminated. A typical example of user participatory design is a project called the Flatwriter by Yona Friedman in the 1960's (figure 24).8

Those who choose to design their homes with DIY software typically create homes designs that are infeasible or plagued with problems because they lack the design expertise and are only directly involved in 1/3 of the design process in 1960 (figure 6). This is startling considering more than 2/3 of the country lives in detached single-family homes (figure 6). The reasons for this vary a great deal and are only directly involved in 2% of single-family homes in the U.S.

The Flatwriter was a clear example of user participatory design in which the future inhabitant of an apartment building selects from a series of formal design options in order to create the flat they would eventually live in. In this scenario, the architect is responsible for creating the experience of possible solutions for the user from which to select. This project will be detailed later.

Those who choose to design their homes with DIY software typically create homes designs that are infeasible or plagued with problems because they lack the design expertise and are only directly involved in 1/3 of the design process in 1960 (figure 6). This is startling considering more than 2/3 of the country lives in detached single-family homes (figure 6). The reasons for this vary a great deal and are only directly involved in 2% of single-family homes in the U.S.
A proposal for a digital application that unites the architect and the occupant in the design process of a home by transforming how design criteria are obtained and controlled.
an alternative to the way design criteria is currently gathered

- and -

an advancement to the current DIY home design software and floor plan catalogs

Figure 2. Examples of the software and 3D models used to create floor plans for at-home DIY software packages as well as examples of catalog home designs.
Economics and efficiency of detached single family home development

A lack of understanding by the public about the benefit architects bring to a project beyond aesthetics.

Proposed solutions for the perception of industry problems:

- Increase efficiency and reduce cost of design
- Create a broader, less privileged relationship
- Design control stays with the occupant/user
- Direct communication between architects and occupants
- Educate the client during the design process
In the 1960s and 1970s, many philosophers did research into user-participatory design and architectural computing. Among those whose work is most noteworthy as evidenced by their continued relevance in contemporary discussions, are Nicholas Negroponte, Christopher Alexander, and Yona Friedman. Each of these architects developed methodologies that utilize technology and computing as the armature of a design process and to some extent, propose a design system that favors analytical and/or logical thinking over intuitive thinking. Each methodology creates a technological platform from which to work and results in a sample of differentiated projects that are each realized by implementing varying degrees of end-user participatory design. As one who – beyond the realm of this thesis – is studying, participating in, and actively developing a system in which computing and technology play a significant role in the design process, I am intrigued by their distinctive conclusions, specifically as they relate to the roles of the end user and the architect.

The technology- and computing-based design processes developed by these three architects are valid and process conceptions that have influenced the evolution of technology and computing over the past five decades and are still seen as they would decade earlier.

I will begin this argument by providing an overview of each author and their respective theories as derived from their written work. Relating to the use of technology and computing in the design process, the narrative will be limited to those authors whose theories have been influential and whose work on computing and architecture regarding the use of technology and computing in the design process. Through this structure, I will pay specific attention to the proposed design process that resulted from the theoretical argument as well as the realized projects.

I will begin with and spend more time discussing Yona Friedman’s work in order to provide clarity to the computational architectural design process, the backbone of which can be applied to the subsequent theories.
In Toward a Scientific Architecture, "Friedman’s main objective is to ‘democratize’ design, to free the user from the ‘patronage’ of the architect, to enable ‘non experts’ to make their own designs, as they are the ones who better know their needs and desires and, most importantly, bear the risk of failure."

Friedman argues that in the past, architecture consisted of a "simple chain of operations" in which the architect worked directly with the client and future user. In its most basic form, the future user makes decisions directly about the finished product. However, he argues that as buildings became more complex, the architect became involved in the process. The future user conveyed his specific needs for his building directly to the architect who, in turn, translated the future user's needs into the design of the finished product. In this arrangement, the architect essentially does not exist in the decision making process. The architect was the middleman between the client and the builder but "all the decisions had been made exclusively by the client." What has changed in the present (1960’s) is that the architect now works for thousands of future users and it is thus impossible for the architect to consider all of the needs and requirements of every future user when designing the building. He argues that the industry is left with two solutions:

1. Supply a large enough number of architects…so that each of them can devote himself to a very few clients.
2. Reduce the period of time spent gathering information (between the client’s visit and the construction of the hardware)"

Given that it would require an unbelievable number of architects to make option 1 feasible, the industry has chosen option 2. The result is that instead of designing for each individual user, architects now design for the specific needs of the average future user.

The problem with this approach, he states, is that the average user does not exist. To express this in an extremely simplified manner: the architect has gone from designing for one user, to many users, to designing for no one.

Friedman states, “The act of deciding also implies that the one who makes the decisions is the one who takes the risks. Any system that does not give the right of choice to those who must bear the consequences of a bad choice is an immoral system. However, that is exactly the way that architects and planners work. They make the decisions and the users take the risks.”

Through this process, Friedman recognizes that the future user must thoroughly understand the risks involved in making design decisions, stating that it is "essential and dangerous to make choices to people who have not been properly informed about the consequences of their decision.” Friedman argues that the role of the architect should be to construct the “framework of a building which will allow the user to make an informed decision of designing the spaces in which they occupy.”

This thinking paved the way for Friedman’s Flatwriter a hardware and software solution that allows the end user or occupant to design their housing unit to their exact specifications by using a combination of output devices and computer software that can be programmed to suggest arrangements of programmable spaces in relation to the user's needs.

To accomplish this he diagrams possibilities of connections between exterior spaces to interior spaces and interior spaces to one another. The following are a series of diagrams originally drawn by Friedman that I have redrawn here for clarity (figures 10, 11, 12, and 13).
Like Bénabou, Alexander also proposes a systematic process to organize and organize their solution to a specific problem. In a basic design problem, the designer must first generate a list of requirements that the solution must meet. This is followed by the development of a solution that meets all of these requirements. However, in ubiquitous computing, there is often a simple way of finding a suitable solution. This is because, in general, the requirements are not independent and can often be combined into a single solution.

Figure 16 is Alexander's diagram of the concept: the problem-solving process. The problem, represented by the nodes, is interconnected and there are dependencies that must be considered. The solution, represented by the arrows, is a series of transformations that must be made to the problem to arrive at the desired solution.

Therefore, there will always be one or more routes (Figure 17) that allow for independent adjustments. Here, making adjustments to the nodes in the problem can be made independently of every other node. In making the problem in a more complex manner, the number of independent adjustments increases. This can be done through a process of substitution, as shown in Figure 18, where each of the nodes in the diagram can be a "set of relations."

The starting point of a set is the beginning of the diagram. The point of substitution is the substitution of the starting point of the diagram, which is a simple alternative to the initial concept. The process of substitution demonstrates the power of the concept of substitution, as it can be used to create a new solution.

Figure 19 is a diagram of the process of substitution, which has been utilized numerous times. In this case, the process of substitution is used to create a new solution that is different from the original solution. This demonstrates the power of substitution and how it can be used to create new solutions. Overall, Alexander's work has been influential in the field of design and has been utilized countless times.
In referring to Nikesign, I will be focusing my analysis on the work of Nikesign’s Christopher Grimes, as his work on the Nike Air Max 97 and Nike Air Max 2017 exemplifies the interplay of desire and futurism with the use of materials, the potential for interactive experiences, and the normative practices that can be employed in design.

The NikeAirMax 97 was launched in 1997 and was designed by Tinker Hatfield. The shoe was a significant design in the history of Nike, as it utilized a new technology called Air Max, which provided better cushioning and shock absorption. The shoe was also notable for its unique aesthetic, with the use of different materials and colors. It was a popular design, and it remains a classic in the world of sneaker culture.

The NikeAirMax 2017 was launched in 2017 and was designed by Nikesign. The shoe was a continuation of the NikeAirMax line, but it featured several new technological improvements, such as the addition of a new type of cushioning technology. The shoe was also notable for its unique aesthetic, with the use of different materials and colors. It was a popular design, and it remains a classic in the world of sneaker culture.

The NikeAirMax design philosophy emphasizes the use of materials and technologies that can provide a better experience for the wearer. The use of these technologies is not just about creating a new shoe, but about creating a new experience for the wearer.

The NikeAirMax design philosophy also emphasizes the use of materials and technologies that can provide a better experience for the wearer. The use of these technologies is not just about creating a new shoe, but about creating a new experience for the wearer. The use of these technologies is not just about creating a new shoe, but about creating a new experience for the wearer.

The NikeAirMax design philosophy also emphasizes the use of materials and technologies that can provide a better experience for the wearer. The use of these technologies is not just about creating a new shoe, but about creating a new experience for the wearer. The use of these technologies is not just about creating a new shoe, but about creating a new experience for the wearer. The use of these technologies is not just about creating a new shoe, but about creating a new experience for the wearer.
Looking at these projects the way that I have, I see very much as a mathematical programming method that fits constraints, but it was not a computing model that was made for constraint-based work. Instead, it was made for operations that those architects knew they wanted to create from the drawings.

To elaborate on two terms I just used, mathematical programming and constraint programming. With mathematical programming, a predefined sequence of steps is defined to reach a particular result (Figure 38).

While there are similarities described above, the two solutions are presently aligned and the design can only be considered “finished” when the sequence of steps is complete. This means that the end product is predetermined before beginning the design process. This is problematic because architectural design is not a mathematical. The design process is not a prescribed sequence; it is iterative and non-linear. The solution to a design is not the end result but rather some desirable point in the process when we feel that all our criteria have been met. “We do not find the solution to a set of design specifications; we find one solution out of many alternatives.”

Constraint programming differs from mathematical programming and akin to architectural design in that it is non-linear and therefore does not specify a sequence of steps. A constraint is a rule. As the name suggests, constraint programming considers any number of constraints or rules, which must all be satisfied to achieve a solution. Thus any solution can be evaluated against any number of constraints.

Regardless of the number of constraints and how they are related, there is still a possible solution. As Alexander suggested, finding the solution is made by compromising between any number of elements within a given solution space.
Constraint Programming

Constraint - 50% of box must be orange

Solutions:
In the context of architecture, the
problem is enhanced by the fact that population
at large is unaware of what architects do, what
is considered when designing a building or why
we consider all of that information.
To formulate a more accurate way of collecting
information from clients I researched the
science of survey methodology. Within this,
there are three main prerogatives: question
wording, question order, and question typology.
In summary there are ways of asking certain
types of questions that are worded in a
particular way and presented in a particular
order that have the greatest potential of yielding
accurate and truthful responses from the future
occupant of a home.
The BVO model employs constraint (CP) instead of mathematical programming (IP) to determine a set of constraints that must be satisfied to achieve a solution. A different programming paradigm and that was used herein is OPL (Optimization Modeller Language), which specifies a set of constraints that must be satisfied to achieve a solution. The variables form the basis of constraints, and objectives, the parameters will not go back and forth during the design process. The role of the architect in the architectural design process, besides the obvious ability to generate creative processes, is also the ability to provide constraints, and objectives, the parameters will not go back and forth during the design process. The BVO model is a promising tool in the framework of a modern system.

In attempting to determine what occurs in the computational process, I reduce to the problem of a cost-effective building. For implementation, the BVO model facilitates the development of cost effective buildings, but it is good for an exclusively architectural and engineering problem. Arguing in an opposite fashion the architect utilizes logic and set theory to determine the best fit for design decisions, whereas variables can be used as inputs for the computational process. The use of constraint combinations, for example, by allowing the opening ratio to use a range between 0 - 100 %, the search space increase when the specifications of decision variables are inconsiderably high. For the visualization of its results, the BVO model is geared for use exclusively by architects and engineers. The BVO model is a promising tool in the framework of a modern system.
Architectural design is affected by rules. Rules can be used to constrain the solution space. Computer can extend the number of feasible options.

Design solution has to meet specific requirements (Constraints). The design has to strive for specific goals (Objectives). There are choices available that meet the constraints and objectives (Design Variables).
BUILDING-VOLUME OPTIMIZATION

A computational decision-support for designers addressing problems associated with missing quantitative design aids during the early architectural design phase.

Minimize: \( LCC_{PV} = CC_{PV} + EC_{PV} + OMC_{PV} + RRC_{PV} \)
Genpod variation results

Visual catalogue of generated alternatives: A-total grade, B-extrinsic criteria's values, C-deviation/adherence to performance envelope (intrinsic criteria). D-generated alternatives (the alternative with the highest grade in the current fitness settings is marked by a dark/red rectangle).

Figure 3: User interface, initial setup and generated alternatives visual interactive catalogue (the alternative with the highest grade in the current fitness settings is marked by a dark/red rectangle).
It would be impossible to analyze every element of design and critical decision in the design process, therefore the scope of my argument will specifically consider a finite set of three variables: (1) site, (2) enclosure, and (3) materials. These three variables are the topics outlined by David Leatherbarrow in his book, The Roots of Architectural Invention. In that text, Leatherbarrow states that every architect must consider site, enclosure, and materials at some point in the architectural design process. Moreover, he argues that these elements must be worked out fully before considerations of style and aesthetic are applied.

The interface I have designed is a constraint-driven program that (1) indirectly obtains missing design information by asking the user about the quality of the spaces in their home in layman’s terms; (2) it serves as an impersonal communication interface between the future occupant of the home and an architect; and (3) it is 3D modeling and design software that uses the information that it gathered to facilitate the design process.

The most troubling aspect of current DIY home design software is the user is typically unaware of the full scope of their design decisions. This lack of consideration is most problematic when building code violations or repairs, or building code restrictions that hinder a passively-cooled design that was the user’s primary motivation. The primary motivation of the interface is to collect necessary and accurate data by rendering the needs of the architect into questions that are comprehensible to the user. This is necessary because the nuances of building a passive home can be too different between an informed and uninformed user. It is client’s job to determine the best design for their home, not decisions that go into making that design feasible. The information gathered by the program can aid in determining whether a change is needed to nudge the user towards the design that they thought they wanted versus the one that can avoid code violations and repairs.

The primary objectives of the interface are to collect necessary and accurate data by conditioning the needs of the architect into questions that are comprehensible to the user. This is necessary because the nuances of building a passive home can be too different between an informed and uninformed user. It is client’s job to determine the best design for their home, not decisions that go into making that design feasible. The information gathered by the program can aid in determining whether a change is needed to nudge the user towards the design that they thought they wanted versus the one that can avoid code violations and repairs.

The interface begins by requesting simple, objective information such as (1) the address, which contains design information by asking the user how they want to live; the architect should facilitate the most appropriate design to meet that expectation.

The occupant is the expert in how they want to live; the architect should facilitate the most appropriate design to meet that expectation. The program can formulate that question by defining constraints that define the maximum buildable volume of the house.
Client/User View

Architect View

68.

69.
Continue to work in the software through completed construction drawings.

Download constraints and graphic information in order to take them to a builder or architect to complete construction drawings.

Work without architect support similar to typical DIY software while utilizing established constraints.
1. Nathan Aleskovsky is the founder and CEO of ShowCode LLC – a software company that develops software solutions for code enforcement, building code and regulatory compliance analysis.

2. (LaBarre 2008)

3. (U. S. Census Bureau 2013)

4. (LaBarre 2008; U. S. Census Bureau 2013)

5. (LaBarre 2008)

6. (Thompson 2012)

7. (Vardouli)

8. (Friedman 1980)

9. (Vardouli)

10. (Friedman 1980)

11. (Ibid.)

12. (Ibid.)

13. (Ibid.)

14. (Alexander, Christopher 1964)

15. (Leatherbarrow 1993)

16. (Ibid.)

17. (Ibid.)

18. (Ibid.)

19. (Ibid.)

20. (Ibid.)

21. (Ibid.)

22. (Varnum and Leal 1977)

23. (Ibid.)

24. (Ibid.)

25. (Ibid.)


27. (Ibid.)

28. (DeWalt, Keyes, and Capello 2009)

29. (DeWalt, Keyes, and Capello 2009)

30. (DeWalt, Keyes, and Capello 2009)

31. (Ibid.)

32. (Ibid.)

33. (Ibid.)

34. (Ibid.)

35. (Ibid.)

36. (Ibid.)