Bespoke Tools for Contemporary Design Solutions

Ralph Spencer Steenblik & Will Wang

1R. S. Steenblik
Rsteenbl@kean.edu
W. Wang
wwang@pcparch.com

Abstract. This paper explores the process and importance of designing and implementing bespoke toolkit solutions within the architectural design discipline. Along with the need for bespoke design solutions comes the need for fluency in architectural principles, digital tool facility, and computational development skill sets (the combination are, today, are still an uncommon skill set). This skill set combination, quite possibly, will become increasingly necessary for design teams to incorporate. This paper argues, through a series of case study projects produced by an internal platform; that the way forward for the architectural design discipline is through bespoke tool-sets geared toward meeting the needs of architectural designers. Design teams are pursuing increasing levels of sophistication and intelligent solutions that meet the demands of problems faced in the building industry today.

Keywords: BIM; Data in design; Custom workflow; Facade; Paneling; Design computation

1 Introduction

Design teams are pursuing increasing levels of integrated intelligence into the final design product in an effort to better participate in a cultural dialog, informed and informing humanity. By staying contemporary on the process of making intelligent tools, one is able to expertly maneuver the design process to produce optimal solutions; this requires a continuous ingestion of innovative computational methods. Architects not only avoid design reductions due to technical inadequacies, but also gain higher levels of agency in affecting the design product. Like many peers, our studio has a research arm specifically focused on the future of the practice. It is an internationally distributed platform, named REACH, charged with integrating innovation into the design process. The group is involved in analyzing, and designing intelligent and informed software tools for designers. Our approach in making these tools is a similar design process to that of designing buildings. These toolkits are built upon the principles of building/city information modeling. Our questions are specific to the tool, but in general terms, we seek a more streamlined building design process through computational assistance. Moreover, we are actively integrating machine intelligence and informed databases to help us create more sensitive design solutions.
2 Background

The profession now faces a landscape requiring a whole suite of digital tools to produce a piece of architecture with any level of intelligence, complexity, sophistication, or significance. There are interesting examples of a single digital tool being too heavily relied upon, within the built environment; creating designs which are too closely predicated on the underlying logics of the tool without adequate design process. On the other hand, with the departure from a singular heavy reliance on one tool as the primary design methodology comes a familiarity with a multitude of software tools, with an intimacy that makes one acutely aware of the challenges, and shortcomings of each tool in time. This creates an interesting conundrum, passively slog through the limitations of the existing off the shelf tools in a customized design process, or invest resources to develop intelligent, informed and bespoke toolsets, attempting to meet the needs of the design processes of a particular studio culture and quite possibly proving to be useful for an entire industry. (Wang, Steenblik 2019)

One challenge is limitations in the pedagogical process of educating architectural designers. Literacy in computer science, robotics, and related fields, a prerequisite in order to address the problems previously stated. Yet the opportunities and toolsets requiring this literacy to be actively taught within the basic architectural education, in part due to the already full curriculum requirements. Thus additional, supplementary, or supporting education is required in order to provide these needed skills in order to actively participate in these conversations, making it a specialization beyond the traditional design practitioner’s skillset. Making off the shelf digital design tools nearly the only option for most practitioners. The outcome have a likelihood of being too highly influenced by the logics of the particular softwares employed, and in turn produce solutions that do not respond holistically to the cultural issues of our time.

Off the shelf digital design tools are no longer sufficient to adequately address contemporary challenges within the architectural design process. Building Information Modeling (BIM) was developed as a way to move passed the traditional digital modeling methods. Yet even BIM proves to have major limitations. Some BIM software tools, for example, become a proprietary black box, so much that the designer loses intelligence on the geometric properties. This can cause significant modeling resources by inefficient trial and error.
The REACH team has several case studies which help to solidify the importance of intelligent, informed and bespoke tools. Recently the platform has been working on several large, resource intensive efforts resulting in toolkits made to work with existing software packages already used in the studio. As was introduced earlier these tools suggest a modal shift in the practice away from a lack of “...in-depth analysis, adaptation, and evolution of [designers’] work” (Celanto 2007), towards one where “...analytical models...coexist simultaneously with parametric models, so that they are immediately usable” (Aguiar et al. 2017, 29). Or in other words a shift toward intelligent, informed and bespoke information modeling solutions. Beyond greater iterative and analysis capabilities, developing these toolkits pushes the knowledge base from heuristics that get lost with staffing changes, into algorithms internalized into the workflow for an ever increasing repository the studio can build upon; we present evidence toward this argument below with the following case studies: 1. REACHTower, 2. REACHCampus and REACHUP 3. Customized Toolbars. These case studies are introduced in the methodology section and expounded upon in the results section.

3 Methodology

The concept, fundamental to our team’s process, is design data: harvesting BIM data and the utilization of “softBIM” strategies. This concepts can also be characterized by the terms intelligent, and informed software solutions. In the most concrete way, the main focus of the team's efforts are to create tools with an interface developed to integrate data into digital 3D models, quickly informing designers of solutions with “fingertip” accessible analytics, and empowering them to achieve more effective solutions. Some of these tools are highlighted in this paper. Without a meta-dataset focused on the design practice and the design process of past projects, the ability for computational aid is minimal; thus emphasizing the collection of metadata also becomes pertinent.

Our bespoke workflow, wrapped in a user interface called REACHTower, keeps architects informed each step of the way in terms of the balance between feasibility and aesthetics of design while attaining precise and rapid documentation. REACHUP is an urban planning design assistant, organizer, and interface for complex objective representations. The tool allows designers as well as clients to see the strength of BIM even as early as the Schematic or even conceptual design phase within the urban planning process. It combines 3D digital models with a database by providing live queries and instant feedback loops on modeled geometries. The tool enables decisions to be made based on any number of complex design parameters presented all in one place, in a succinct way. A similar project dubbed REACHCampus, has been created for use in higher education planning. Additionally, the team has worked on more traditional toolbars in several design software packages. From this work has grown some innovative solutions applicable to the architectural design disciplines.
3.1 REACHTower Part of REACHTower is a paneling exercise which expresses the custom workflow previously mentioned, mediated by a proprietarily crafted user interface built on industry accepted architectural modeling tools. In early stages of the project, the design team has tried paneling in more singular software environments but that only highlights the limitations of the standalone softwares themselves. Even with readily made plugins such as Paneling Tools 1 and Hummingbird2, the risk of discontinued external support runs high. A big advantage of our method is the controlled accuracy of interoperation. Nate Miller (Proving Ground) has written about the problem of different geometry generations between Revit and RH.3 In our workflow the data transferred are all point coordinates. There is no interpolation, and decimal precision can be prescribed by the user.

Our method for façade modeling, deployment, and documentation is maturing. We have pushed it far enough that the most daunting aspects of making design changes (i.e. modeling) is mostly computing time per se. The interoperability workflow is only one custom, integrated, tool built into the REACHTower toolkit. Additional tools include an industry collaboration with a software company working on VR sculpting. This collaboration allows the toolkit to have a greater life of its own beyond the singular modeling environment, as was previously discussed as an original impetus for the teams’ conception in the first place.

![Facade Panelization Workflow with REACHTower](image)

---

1 Developed by Rajaa Issa at McNeel Associates. See reference here https://wiki.mcneel.com/labs/panelingtools
2 Independently developed by Tim Meador and Mario Guttman. See reference here https://ghhummingbird.wordpress.com/about/
3.2 REACHCampus and REACHUP
In developing REACHCampus the focus was on overall strategies over particular formal articulations. Schematic massing studies should come later in the process and in fact at a lower resolution. Therefore a large amount of overall campus information was given, to extrapolate guiding principles. Data includes energy uses, campus transportation, well-documented tree species (including GPS locations), etc. It becomes static to only look at each category on its own, and difficult to keep track when multiple categories are juxtaposed to study connections or influences. REACHCampus is a prime example of leveraging data to increase designers’ access to intelligent and informed tools.

Figure 2 Built-in workflow logic, optimal for computer/human dialog

One of two main features of the REACHcampus toolkit aims to provide a more dynamic query mechanism than is currently used in the industry (printed reports and on-screen text files). We use two kinds of data. One can exist entirely independent of a 3D digital model, and the other is actually modeled in the digital modeling environment RH. The other main feature of REACHcampus contains most of the computing. It searches through the database, matches entries with 3D model elements, and calculates relevant statistics and analytics.

A distinct phase of master-planning called data gathering, often at the beginning of the architectural masterplan workflow, involves the painstaking process of extrapolating as many diagrammatic representations as possible. These diagrams are then compiled in a hefty printed medium as a design reference. Digital copies of such material only exacerbates the deluge of un-synthesized information a designer is expected to digest, as concurrent visibility of multiple categories of information becomes clumsy under most circumstances. A more flexible way of data representations is due.

REACHcampus is comprised of a user interface (UI) and back end algorithmic collection connected to a database specific to a particular project. The database in this data-driven process has been termed “DesignData” by the authors of softBIM: an Open-ended Building Information Model in Design Practice. “Through associating raw data with objects, meaning is given to the data, thereby turning it into useful information for the design process.” It consists of simple data visualization interface components which express the queries of the back-end database and link them to 3D
elements. This combination extends the functionality of 3D geometries beyond the scope of its traditional, native CAD software. The majority of REACHcampus’ functionalities also apply to urban planning, because college campuses are either in urban settings or creating their own “urban” (National Geographics, n.d.) environments. The toolkit is geared towards university master plans, where complex design objectives and large collections of iterations can be rapidly represented in an organized fashion empowering designers to make more informed design decisions more rapidly.

On-the-fly optioning tools are integrated into our interface for near real-time feedback. Similar “reflexive” design methods have seen increased implementation on the object level. In Towards an Architecture of Cyber-physical Systems, Kathy Velikov and Geoffrey Thun furthered the idea of “…placing the act of design within the highest order of system hierarchy, being simultaneously purposed and reflexive.” (Velikov and Thun 2014, 333) REACHcampus brings the information feedback loop to larger scales and reduces myopic planning in isolation. With solver integration we are able to optimize for desired parameters. This gives the designer benchmarks and intelligent cues from which to begin the design process in a much more informed way.

A key metric in master planning is area calculation. A traditional approach to acquiring this information is through 3D-modeling surfaces or polygon curves that represent building floor plates (CAD software often have a command that calculates the area either as an aggregate or as individuals). After the area has been calculated it must be manually entered as text values, updated with every design revision affecting areas.
With REACHcampus, as is common with BIM solutions, building level information is stored in the database as text strings. The interface reads them, matching buildings via identifiers, and produces floor plates dynamically. In REACHCampus there is no modeled geometry of the floor plate unless desired, keeping a leaner digital model optimized for performance. A change in level elevations in the database is precisely reflected in the model upon a new query. This eliminates the need of human tracking, which is often error-prone, as well as automating the calculation process.

Once area is calculated, many related key metrics can be computed such as floor-area ratio (FAR), daytime population, and estimates by unit cost, etc. These algorithmic procedures are also built into the interface so the designer can quickly see results with a toggle, or can even edit the definition of the parameters in the equation. Although the math mostly involves common arithmetic operations, the expedience afforded by an automated, simplified interface improves experience by pushing metrics right to the designer’s fingertips.

The strength of our toolkit’s metrics computation is flexibility. Because the specialist has access to the back-end of the tool, it is highly customizable for creative use or adaptations. Besides existing conditions and their representations, we have created a massing generator capable of repeating the same marriage between metadata and geometry. Even though a designer is more than competent to model in the native RH interface, the massing generators and their UI enables interweaving options into the script computation – the result is DesignData.

![Figure 4: Integrated massing generator primed for algorithmic analysis](image)
In order to produce the organic effect of city growth, we embed a pseudo-randomization algorithm into the REACHCampus massing generators. Due to the nature of the visual scripting environment’s computation sequence, the common practice in producing pseudo-randomization is through a “seed” controller. In other words, there is a known one-to-one relationship between a solution and an input seed number, albeit solutions seem arbitrary to each other. This phenomenon runs against the concept of randomization. So we decide to iterate through options in a completely unpredictable and ephemeral manner. To compensate for the lack of indexing in this arbitrary process, we have to create our own temporary storage in computer memory where solutions can be deposited upon a toggle switch. This way we can retrieve an option.

Anecdotally, this mechanism applies a paradigm of machine learning in a way that is uncommon within architecture. There has been some difficulty for designers to integrate true machine learning processes into the design process, because of the “learning” period. Most designers are able to find the solution during the time it takes for the learning to take place. With the use of the analytical tools and the stored solution sets we are able to cross beyond the learning barrier prior to first use by an end user, and as more users engage the tool the so called AI will become smarter. This breakthrough has the potential to fundamentally alter the method of working within the firm if not large portions of the industry.

Figure 5 Isovist, integrated as a part of our tools to study site-lines
3.3 Customized Toolbars We have observed that one derivative, in the process of researching new processes or techniques, may be studio-wide increases in productivity through implementation of these new methods and solutions. By understanding software API’s, a collection of functionalities specific to the studio can be made to a customized toolbar. Take stairs for example. They are a very familiar object to architects but rarely a concern for naval engineers or jewelry designers. In dedicated environments for architects, there are stair tools readily available. But in freeform environments, the architect has to model the object from scratch. With scripting, we can make a parametric tool that sits next to all the out-of-the-box toolbar buttons. Over time, the effort saved adds up to greater efficiency.

Other outgrowths of the bespoke tool creation have been specific functions manifested as toolbar commands, within the different software packages used within the studio. Sometimes these tools have been the predecessors of the custom interfaces and other times they have been independent. Regardless, we have been able to incorporate many features into our toolbars which make them very appealing for the productivity of the studio.

For example while creating the urban/masterplanning exercises we were able to produce many tools with standalone application within the studio outside of the specific custom interface. The native IsoVist tool inside of RH is primitive and limited in utility by itself. With simple, strategic embellishment, tools such as the IsoVist can become very powerful visual tools. Properly automated, the tools are then quickly integrated into the everyday analytical toolkit. After innovating on top of the existing tools available in the platform they become new tools in their own right. In the case of the enhanced IsoVist tool, for example, has become something other, called View Lilly.

![Figure 6: Analytics for the tool provide feedback on tool usage](attachment:image.png)
After implementing such a tool in the design studio we are able to begin gathering data on its use. This allows us to mold the tools and the collections of tools in ways that are more intuitively useful to designers’ ways of working. Additionally, these datasets point us in directions that are useful and point to holes in the custom tools not yet implemented. Beyond the data on the actual tools themselves we are most interested in capturing data on the designs the tools are used on. By gathering this data we can begin to understand more about the spaces that come across the studio drawing boards. After time the data sets will gain robustness toward more acute solutions within the design and analytics or informed future projects. Understanding past projects to this level of granular detail is rarely achieved within contemporary architecture studios.

4 Discussion and Conclusion

The territories of intelligent, and informed computational design, can prove elusive, but learning to leverage these modes is key in the progress of the design disciplines and particularly the architectural discipline. The technology and expertise have already arrived. It is a matter of harnessing skill sets in a way that encourages innovation. Young talent is increasingly equipped with the proper knowledge.

Moreover if a design studio can encapsulate its knowledge via algorithms, databases, and automated procedures, they will enjoy not only boosted productivity but also answers to “wicked problems” (Martin 2009, 92). Without the understanding of parametric geometries and efficient documentation, design exercises can easily get lost in the cycles of manual drafting and coordination. Without an analytical dashboard, design exercises can only address very few criteria within a given timeframe. These increases in capability allow us to more acutely begin designing toward solutions to problems such as climate change, measurable marketability, cost and feasibility, increased regulatory hurdles, etc in a way previously unavailable to designers at large.

Bespoke tools are a way of innovation within this design profession too often “dismayed by the elusiveness of success… [and] diminished impact” (Celanto 2007). Computationally intelligent, and informed research through, data harvesting, and softBIM are hopeful ways forward for many of our disciplinary challenges. At the 2017 Thornton Tomasetti’s AEC Technology Symposium, Andrew Heumann and Brian Ringley of WeWork spoke on the impracticality of the single platform fantasy. They emphasized that the industry has shifted into a paradigm of multi-software data exchange.
Having personnel who can make this exchange process transparent and seamless, while integrating levels of intelligence is a necessary asset on any architect’s team. Operating outside of the traditional software platform mode by creating intelligent, and informed customized tools, we are able to increase the freedom with which one works by removing preconceptions upon which proprietary software packages are predicated, and by operating based on our own terms. This in turn allows us to address cultural zeitgeist in a way that is simply impossible to do when tethered to predetermined logics. Architecture is inherently top-down, regardless of our efforts to incorporate bottom up processes. Architects therefore must gain the highest level of intelligence and understanding of their craft.

Acknowledgements We would like to thank Pelli Clarke Pelli Architects, particularly Rob Narracci and Fred Clarke for seeing the vision. This paper builds on previous research found in “Intelligent & Informed” Vol. 2, (2019), 111-120. This paper focuses more on the cultural ramifications of these custom workflow based computational tools.

References
