# Software Development Within Architecture: A Graphical Evaluation and Discussion of Its Trends

**C. Grey Isley[1], Dana Gulling[1]**

[1]North Carolina State University, Raleigh, NC

ABSTRACT: Developmental leaps within digital technology has impacted architectural form and advanced how architects communicate, analyze, and incorporate advanced building technologies into their designs. Occurring over a period of decades, software's impact on the practice of architecture has escalated through an increase in its accessibility and adoption by the profession's leading architects. This has resulted in digital technology becoming one the largest contributors to innovation within the architecture, engineering and construction (AEC) industry, fundamentally changing the architectural process, and to some extent the contemporary design language. To better understand the relationship between digital technology and architecture, this study looks at a sampling of software utilized by the industry and evaluates it over a spectrum of time based on functional and developmental characteristics. Through the creation of a graphical representation of the collected data, patterns between taxonomies, software development, and its usage within architecture have been observed. It is proposed these trends can aid in the understanding of the landscape of software development, how it has transitioned over time, what programs are available for usage within architecture, and how they are interrelated with the architectural process. These trends -- aided by the understanding gained from their analysis -- can then be utilized to facilitate a discussion regarding how the trends relate to larger developmental tendencies within the AEC industry and be used as measures for the changing landscape of the architectural process.

KEYWORDS: Architecture; software; innovation; software development; digital technology

## INTRODUCTION

The impact of digital technology on the practice of architecture over the past 35 years has been significant, contributing equally to the evolution of the built form and the design and construction process. Arguably as revolutionary to the design process as the introduction of paper or the discovery of perspective drawings (As and Schodek 2008; Kvan 2009), the era of the computer has fundamentally altered the way architecture is practiced through the introduction of advanced computational-based design methods, digital fabrication, simulation and analysis, digital representation, and advanced delivery and construction methods such as building information modeling (BIM). This redirection of the process has set a new mandate for the discipline, changing the design process from a representational and paper-based system to a digitally-managed and influenced system of analysis and simulation. As a result, digital technology has displaced more traditional design methods, requiring the architect to develop more digitally oriented ways of thinking and extend a design concept through interaction with a digital system. (Galofaro 1999)

Digital technology's ever-increasing bearing on practice is a result of a surge in access to software that was once almost entirely exclusive to researchers, advisers, and engineers. Related to both technological advances and economic factors, this access has resulted in a proliferation of possibilities as architects can now directly connect to and integrate simulation, optimization, building information systems, manufacturing process, visualization, and complex forms into their designs. (Grobman 2012) As a result, some relate this digital shift to an evolutionary movement that will establish a new era within architecture. (Aksamija 2016) In consideration to these factors, this study is interested in the potential relationship between the development of digital technology and the practice of architecture. As such, it is considered that patterns within the development of digital technologies can be used to draw conclusions that relate to trends within architecture, and the interconnectivity of software, software development, and the architectural process.

To aid in understanding the relationship between digital technology and architecture, a sampling of software within the architectural community was evaluated based on significant developmental information and function. The collected information has been graphically represented along a timeline illustrating the selected software's development and mapping their primary and secondary functions. Utilizing this software development graphical representation (SDGR), patterns were observed between the study-defined

taxonomies, software development, and usage within architecture. These trends can aid in the understanding of the landscape of software development: how it has transitioned over time, what types of programs are available for usage within architecture, and how they are interrelated with the architectural process. Through this understanding, the patterns observed in the SDGR can be utilized to facilitate a discussion regarding how these trends can relate to larger developmental movements within the industry and be used as measures to understand the changing landscape of the architectural process.

## 1.0 Methods

Through literature review and discussion with practitioners, the study generated a sample of 57 software programs associated with architecture design. This research did not intend to include all software commercially available and therefore inclusion was based upon reference or discussion of the software within the AEC industry, known use in practice and academics, historical importance with respect to architectural developments, trends, or other software, and the availability of historical data. It is acknowledged that software or functions of the included software may have been omitted due to a lack of available information regarding the software or other related factors. Any omission is not considered significant for this study as the focus is on developmental patterns and not on the specific software.

For the utilized software packages, data regarding development dates, manufacturer, current version number, primary and secondary functions, associative software, and industry of origination was collected, when available. This data was retrieved from various sources including manufacturer-provided documents, academic publications, archived websites, and journals. In cases where an exact release date was not available, the developer's date of establishment as an entity was substituted when appropriate. When there were conflicting dates of release between sources, the sources were considered with developer-generated material taking priority, then peer-reviewed material, and finally publicly-generated sources such as blogs or website articles.

### 1.1. Software Data Graphical Representation (SDGR)

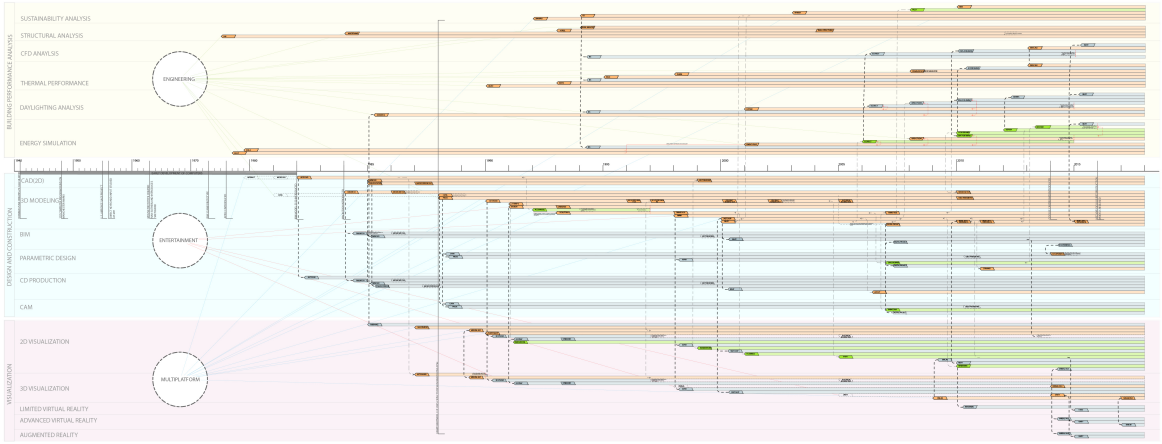


**Figure 1**: Software Development Graphical Representation (SDGR) (download: go.ncsu.edu/sdgr)

In the SDGR, Figure 1 and downloadable at go.ncsu.edu/sdgr, software programs are graphically represented along a timeline using initial development or release dates, field of origination, first use by architecture (when available), associated software, and key developmental occurrences, such as acquisitions, discontinuation, or significant name changes. The software is organized by three primary taxonomies and their subsequent categories. This structure was generated considering similar classification structures and discussion of the software found in literature. Represented horizontally along the SDGR, the taxonomies frame the larger usage of the software within the industry.

- ***Building Performance Analysis*** *(Yellow)*: related to analysis of building performance and structure
- ***Design and Construction*** *(Blue)*: related to CAD, modelling, and construction
- ***Visualization*** *(Purple)*: related to non-construction document representation

Each taxonomy is futher divided into function categories, specifically identifying the functions of the software.

*Building Performance Analysis*

- ***Sustainability Analysis*** – provides life cycle analysis and embodied energy of materials
- ***Structural Analysis*** – related to the analysis of structure and structural loads
- ***Computational Fluid Dynamics (CFD) Analysis*** – associated with the evaluation of fluid flow within structures and equipment
- ***Thermal Performance*** – performance of the evaluation of heat transfer through assemblies in 2D and 3D perspectives
- ***Daylighting Analysis*** – related to the evaluation of glare and daylighting in a structure
- ***Energy Simulation*** – associated with the evaluation and simulation of energy usage in a structure

*Design and Construction*

- ***CAD(2D)*** – entity or vector-based CAD programs that while can represent in 3D objects are created primarily in a 2D plane.
- ***3D Modeling*** – software used in the creation of 3D digital objects through object-based, NURB, volume-based, surface-based, or similar systems that create and represent objects primarily in 3D.
- ***BIM*** – related to building information modeling systems
- ***Parametric Design*** – provides algorithmic functions that aid in the design or manipulation of 3D objects
- ***Construction Documents (CD) Production*** – used in the creation of construction documents
- ***Computer Aided Manufacturing (CAM)*** – utilized for the creation of documents or coding used in Computer Numerical Control (CNC) systems.

*Visualization*

- ***2D Visualization*** – software used in the creation of static images of architectural representations.
- ***3D Visualization*** – related to interactive or dynamic representation of architecture
- ***Limited Virtual Reality (VR)*** – creation of VR systems that are primarily static and related to lower end headsets such as Google Cardboard.
- ***Advanced VR*** – associated with advanced VR systems that allow for observer movement and interaction often utilizing higher end headsets such as VIVE or Oculus Rift
- ***Augmented Reality (AR)*** – creation of mixed reality experiences that uses either smart phones or dedicated headsets for the blending of real and digital world objects using cameras.

Within the SDGR, software is represented by horizontal bars in the associated function categories, Figure 2. The functions of the software were identified through its discussion in literature and available manufacture information. Software that operates in multiple categories is listed in all applicable categories with the functions ranked into primary and secondary functions. Primary functions, represented as orange, are considered the main function that the program was created to perform or is currently used for in the AEC industry; for example, AutoCAD is primarily used as a *CAD(2D)* program, while Rhino3D is primarily used for *3D Modeling*. Secondary functions, represented as blue, are functions that the software can perform but are ancillary to its designated primary function. Such as, Revit's primary function is *3D Modeling,* but it is also used for *BIM* and *2D Visualization*. The SDGR includes a single primary function for each software, except when the manufacturer or industry's use demonstrates an equal weight to multiple functions. Such as, in the case of SimScale there is little evidence of a hierarchy between its functions related to *CFD Analysis* and *Thermal Performance*.

The SDGR also includes a special designation for software plug-ins, represented as green. Plug-ins are software that function with limited or no capacity without the assistance of another program. For example, Grasshopper requires Rhino3D for full functionality. In cases where plug-ins have multiple functions, the functions are also ranked, with secondary functions represented in blue.

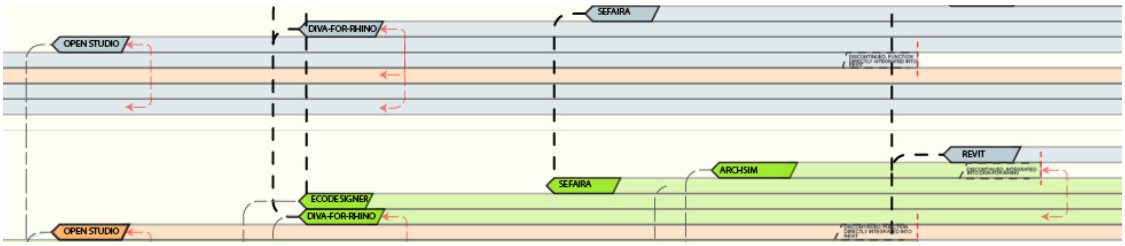


**Figure 2:** Software Representation and Functional Color Hierarchy

Each software bar is preceded by a header listing the software's name at the time of development. Additional headers are located along the bar, indicating significant name changes. The headers are located within the associated year of occurrence along the timeline. If there is an identified difference between the software's introduction date and its adoption within architecture, the difference in time between the introduction and adoption of the software is represented by a dashed bar proceeding the solid bar of the primary function. The dashed header and bar indicates the introduction date of the software and time prior before adoption, with the solid header and bar indicating its adoption and use within architecture.

Connections between a software's functions, along with any relationships between a software and a plug-in or other program, is illustrated using dashed lines. Black dashed lines connect the functions associated with a single software. For example, in Figure 2, the connections between the primary function of Sefaria, located in the *Energy Simulation* category and represented as a plug-in, is connected to its secondary function located in the *Daylighting Analysis* category directly above it through the black dashed line originating from its header. Lighter, dashed gray lines show connections between a primary program function and a plug-in or other related software package. These lines originate from the header of the plug-in or in some cases along the horizontal bar of the plug-in at the time in which they were first associated with the connecting software. In Figure 2, an example of the gray lines originates from the header of EcoDesigner and bar of Sefaria connecting them to ArchiCAD and Revit respectively, which are located elsewhere in the SDGR and not shown in Figure 2. A double-arrowed red line is used to connect the header of a software package acting as an interface (a program that transfers information directly between the user and another program without evidence of the transfer) to its related program. For example, in Figure 2, Diva-for-Rhino interfaces with Energy Plus (not identified in Figure 2).

The software's field of origination is also represented in the SDGR. Defined as the discipline or area of use in which the software was initially developed for or primarily associated with, four categories have been designated with three of them actively represented in the SDGR as a circle. Software corresponding to these areas are connected via a line drawn from the primary function header to the appropriate circle, Figure 3. If a connection is not made to one of the three represented categories, then the software is considered significantly connected to architecture since its development.

- ***Architecture***– software primarily associated with the architecture discipline
- ***Engineering*** – software primarily associated with engineering analysis or the engineering discipline
- ***Entertainment*** – software primarily used for movies, gaming, or other entertainment related function.
- ***Multiplatform*** – software that has been developed or strongly utilized since its release with multiple disciplines and cannot be related solely to one of the other origination categories.

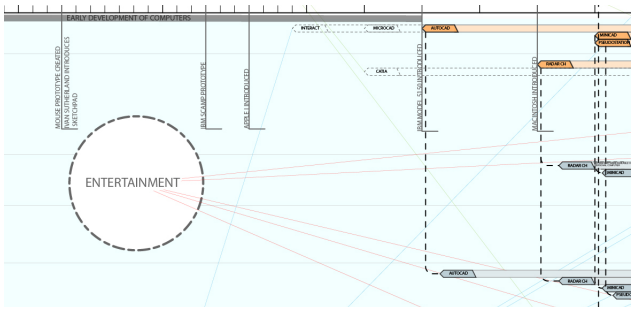


**Figure 3:** Representation of Field of Origination and Connections. All software not associated with a circle (e.g. Entertainment, Engineering, or Multiplatform) has originated from Architecture

## 2.0. Validation through Historical Trends

The evaluation of the SDGR initially concentrated on confirmation of the presence of early developmental trends documented by Aouad et al.(2012), Galofaro (1999), and Kalay (2004) in the SDGR. This consisted of three primary tends: 1) pre-1980 software development, 2) the impact of the introduction of the PC, and 3) the influence of hardware advancements with respect to processing and graphic rendering power. These trends were used to validate the patterns associated with the SDGR.

These trends are outside of the discussion of this paper but are included in summary. First, prior to 1980, software development was primarily related to engineering and governmental funded projects. This trend is present in the SDGR, with the three listed programs originating before 1982, DOE-2, Blast, and SAP, associated with *Engineering* and two of the three developed through government support, DOE-2 and Blast. Secondly, after the introduction of the personal computer (PC), there is an increase in the development of

software in the SDGR, following known patterns. The software increases across the timeline with the number of sampled software almost doubling between the time frames of the 1980's, '90's, and beyond 2000. Finally, as the PC continued to evolve, hardware advancements influenced how software developed. This is specifically seen in the SDGR through development in categories such as *CAD(2D)* and *3D Modeling*. The entity-based systems associated with *CAD(2D)* were more quickly developed due to the ability of early PCs to accommodate their graphic requirements. The *3D Modeling* category lagged in development until PCs were able to accommodate more complex graphics.(Aouad et al. 2012) Outside of these categories the more graphically demanding software (e.g. Unity, Unreal, and Maya) developed at an even later period once graphic hardware was able to support their functions.

## 3.0. Analysis

The SDGR was evaluated in phases for trends associated with: 1) the taxonomy and their associated categories, 2) interactions between taxonomies, and 3) plug-ins. This analysis frames the discussion and its consideration of trends in software development within architectural practice.

## 3.1. Trends within Taxonomies

The study initially evaluated the SDGR for trends within the taxonomies (e.g. Building Performance Analysis, Design and Construction, and Visualization). This evaluation revealed patterns associated with the software's area of origination, function within industry, and the predominant category within the taxonomy.

Trends within the *Building Performance Analysis* taxonomy demonstrate the relationship between a software's field of origination, use within practice, and its development of functions. This taxonomy is closely associated with engineering with 14 programs associated with the *Engineering* category, more than the other two taxonomies. Additionally,19 of the 25 (76%) programs in the taxonomy retain a singular function, with the 6 remaining (24%) having multiple functions. Of these six multifunction programs, five have their primary or secondary functions in *Energy Simulation*, *Daylighting Analysis*, *Thermal Performance* or *CFD Analysis*. The sole software (e.g. IES) outside of this pattern has its primary function located in *Sustainability Analysis* with its secondary functions located in all the above listed categories. These patterns indicate that there is a relationship between function categories, the presence of multiple functions, and the associated software's use within the industry. Such as, it is more likely that a user requiring a software package that performs *Energy Simulation* will also require *Daylighting Analysis,* but it is unlikely they will require *Structural Analysis*. This is associated with the fact that these categories relate to different engineering specialties, such as *Energy Simulation* and *Daylighting Analysis* with mechanical engineering and *Structural Analysis* with civil engineering. This understanding provides insight into the patterns of segregation between categories and the presence of multifunction software. In this case, it can be observed that the specialized nature of the disciplines associated with *Engineering* and the software's use within the AEC industry results in the software of this taxonomy typically not requiring additional functions. As such, this trend indicates that the field of origination and the software's use will directly influence how primary and secondary functions within software will develop.

This pattern continues as the *Building Performance Analysis* taxonomy contrasts with the *Design and Construction* and *Visualization* taxonomies with regards to the presence of multifunctional software, again relating to the software's use and origination. Software in these two taxonomies frequently have both primary and secondary functions. In *Design and Construction* 11 out of the 19 (58%) programs have multiple functions with this number rising to 14 out of 19 (74%) when considering plug-ins. Visualization has less with 6 of the 13 (46%) having multiple functions, with 5 out of the 7 (71%) remaining programs being plug-ins that primarily support programs located in another taxonomy. This inclusion of multiple functions within the software in these taxonomies is an indication that there is a need or demand for multiple functions within these taxonomies and directly relates to their usage and area of origination.

In the case of *Design and Construction* 10 of the 19 (52%) programs are associated with *Architecture*, this increases to 12 out of 19 (63%) when *Engineering* related software is included by assuming these two categories have similar demands for software within this taxonomy. These demands will drive software in this taxonomy to provide multiple functions that allow for both the creation and documentation of designs. For disciplines outside of *Engineering* and *Architecture,* the requirements for documentation may differ and therefore there a reduction in the need for multiple functions in software associated with those disciplines in this taxonomy. Such is the case of 3DS Max, in which it originated outside of architecture and therefore there is no need for functions related to *CD Production* or *CAM*. For this reason, 3DS Max has a singular function in *Design and Construction* while other software in this taxonomy has developed multiple functions to satisfy the needs of their field of origin. This pattern of multifunction inclusion directly relates to the taxonomy and the areas of origination and relates to the trends found in *Building Performance Analysis*.

Finally, *Design and Construction* and *Visualization* contain dominant categories. A dominant category is defined as a category that is comprised of mostly primary functions and indicates its importance in the industry. In *Design and Construction*, *CAD(2D)* and *3D Modeling* are dominant categories as they contain the primary function for 15 of the 19 (79%) programs in the taxonomy. This dominance is strengthened when it is considered that of the four remaining software, two are plug-ins (e.g. Grasshopper and RhinoCAM) that provide access to *Parametric Design* and *CAM* respectively for software in *CAD(2D)* and *3D Modeling*; while the other two (e.g. Layout and Dynamo) are standalone programs that provide similar services. For instance, Layout is a standalone software, but its primary function is to allow Sketchup to generate construction documents. This dominance of *CAD(2D)* and *3D Modeling* indicates their importance to the industry and that they act as the core function in this taxonomy, allowing for the use of secondary functions. For example, production of construction drawings in AutoCAD would not be possible without first creating an object with the *CAD(2D)* function. In this case, the presence of two dominate categories also provides insight into how the development of dominant categories can relate to development trends within the industry. The development of two dominant categories in this taxonomy is a result of the software's handling of object creation and is tied to several factors. This has resulted in the industry using two different methods of object creation and can be tied to preferences and trends in practice.

This association with practice and a dominant category's development is also found in the patterns associated with *Visualization's* dominant category, *2D Visualization*. While noticeable as a dominate category, containing the primary function of 8 of the 13 (61%) programs in the taxonomy, unlike *CAD(2D)* and *3D Modeling, 2D Visualization* is not as pure as it contains secondary functions. Still indicating a primary demand of the industry, this weakened dominance can also indicate a change in industry preferences, along with the influence of other external factors such as PC development. This is similar to the presence of two dominate categories in *Design and Construction*. These links to practice will be discussed more in depth in the discussion as patterns associated with these occurrences can be used to indicate shifts in practice.

### 3.2. Trends between Taxonomies

Excluding the use of plug-ins, there is little to no connection between taxonomies. Software in a taxonomy, with minor exceptions, does not typically have secondary functions outside of their associated taxonomy. Meaning if a software resides in *Design and Construction*, it does not have secondary functions outside of *Design and Construction*.

### 3.3. Plug-in Trends

Generally, plug-ins serve as a bridge between taxonomies. This bridge acts as a conduit for software to incorporate additional functions from outside their primary category and often their taxonomy. An example of this is Sefaria providing Revit and Sketchup access to *Energy Analysis.* This development allows for software to evolve with the industry and provide access to functions that may not have been included in the software but is needed within the industry of use.

## 4.0. Discussion

The understanding of the trends in the SDGR allows for the measurement of current digital trends and the projection of how digital technology may develop in the future within architecture. Additionally, these trends may be used to reflect on how architecture is developing as a practice. This discussion will look at the analyzed trends, expand upon them, and briefly look at how they relate to these two concepts.

Starting with the segregation of taxonomies within the SDGR, considering the influence the software's area of origin has on the segregation and development of functions within a taxonomy, it can be concluded that there is a similar relationship with the segregation between taxonomies and their subsequent connection through plug-ins or functions that cross taxonomies. For example, while software in *Building Performance Analysis* is used within architecture, the software typically originates from *Engineering* and is primarily used by engineers or related disciplines. This results in the segregation of taxonomies as *Building Performance Analysis* is more associated with services that are specialized and outside of those that may be associated with *Design and Construction* or *Visualization*. However, this segregation is reduced through the use of plug-ins and cross-over functions. This is due to the need for the analysis of the models created using software residing in *Design and Construction*. As a result, the connection between *Design and Construction* and *Building Performance Analysis* is greater than that between *Building Performance Analysis* and *Visualization*. This difference is because the goals of software in *Building Performance Analysis* is the quantitative analysis of designs created through software in *Design and Construction*. This does not require the high-level visualization provided by software in the *Visualization* taxonomy. This relationship between the software's use and the field of origination, therefore, both informs the segregation of the taxonomies and how they relate through plug-ins.

This influence continues when evaluating functions that cross taxonomies. In *Design and Construction,* these functions are limited to Cinema4D, 3DS Max, and Revit. This is the result of either the adoption of software from other disciplines or a trend in which plug-ins are developed into secondary functions. The latter of which is associated with Revit and can be related to a change in practice. This will be discussed later along with trends associated with plug-in development. For 3DS Max and Cinema4D, the presence of functions crossing taxonomies is attributed the software's origin. In both cases the software was adopted from outside of architecture and as a result the software has limited functionality within its primary taxonomy, *Design and Construction*. In the case of 3DS Max, it is associated with multiplatform disciplines with its users having differing requirements depending on their area of practice. A common factor is its users will typically require high level visualizations, but not all require the other functions in *Design and Construction*. For example, video game designers use 3DS Max to design game environments, but they have no need to generate construction documents. Driven heavily by their origin, these programs are not capable of providing some of the more industry specific functions associated with the *Design and Construction* taxonomy. This results in the presence of functions that cross taxonomies, but the software may have limited functionality within its primary taxonomy resulting in segregation. This typically results in specialized use within the AEC industry.

Trends associated with a software's origin, as mentioned, can be adjusted through the use of plug-ins. Plug-ins add functions to existing software, both within and between taxonomies, and as a result reduce segregation. As such, their development is linked to the relationship between taxonomies and functions. They can also indicate a taxonomy's importance and adjust for quickly rising trends in the industry. The importance of a taxonomy is indicated through the connections created by plug-ins between taxonomies. For instance, *Design and Construction* is the only taxonomy connected to the other two, as *Visualization* and *Building Performance Analysis* do not have any connections aside from Radiance which stems from its method of creating shadows, which is used in both daylighting analysis and visualization. This pattern of connection indicates *Design and Construction* is more important in architecture, with regards to software use. Furthermore, this indicates the software in *Design and Construction* acts as a hub for the interaction between the taxonomies. Additionally, the rate and sequence in which plug-ins develop between taxonomies can be an indicator of their relationship. Such as, plug-ins between the *Visualization* and *Design and Construction* taxonomies developed earlier than those between *Building Performance Analysis* and *Design and Construction*. This suggests a longer and potentially stronger relationship between *Visualization* and *Design and Construction,* which is indicative of the importance of visualization to the design process.

The ability of plug-ins to adjust for a change in the industry is demonstrated through the more recent development of plug-ins between *Building Performance Analysis* and *Design and Construction*. This growth occurs later than that between *Design and Construction* and *Visualization*. This parallels the increased emphasis on sustainability and energy cost within architecture. This shift in practice has resulted in a need for the integration of sustainable and energy related functions into existing programs in *Design and Construction.* Therefore, plug-ins have developed to provide the needed functions. For example, EcoTect was developed to perform energy analysis within Revit. As demand for these functions has increased, Revit has progressed to the point of incorporating the functions provided by plug-ins as secondary functions that cross taxonomies. This scenario demonstrates how plug-ins may be an interim solution if demand is large enough for a function, but additional study is needed. From this study, the trends indicate this progression can be a result of industry demand, a strengthen of a relationship between disciplines, and a strong indicator of a change in practice.

This use of plug-ins to adjust for growth is not exclusive to functions outside of a software's taxonomy, such as those discussed. Plug-ins can develop within a taxonomy, linking categories. While this occurrence does not necessarily relate to an increased relationship between disciplines, it is a strong indicator of a shift in practice. For example, in P*arametric Design* there is growth over time of secondary functions and plug-ins included in the category. The use of parametrics is a concept that has been associated with architecture since Ivan Sutherland's development of Sketchpad in 1963 (Frazer 2016), however, the advancement of construction techniques in the past 20 years has allowed architects to more easily adopt its use. This has resulted in a wider adoption of parametric design in practice and the growth of secondary functions and plug-ins in *Parametric Design* that link to software with primary functions in other categories. This demonstrates how plug-ins can quickly adapt for an increased demand for functions within a taxonomy.

The final discussion of this study is that changes within or to a primary category in a taxonomy, when prevalent, may also be a predictive measure. This may occur in two ways, through an increase in the development of software within an existing dominant category, thereby furthering its dominance, or through the transition of dominance between categories within a taxonomy. Growth within a dominant category can indicate new or strengthening trends within the practice. For example, there is an increase in development within the *3D Modeling* category, with a decrease in *CAD(2D)*. This change reflects the shift in the architecture industry from 2D to 3D design but does not indicate a total abandonment of 2D design at this point.

The changing of the dominate category within a taxonomy must also be considered. While this has not explicitly occurred within the SDGR, this trend can be discussed using conditions observed between *CAD(2D) and 3D Modeling* and *2D Visualization and 3D Visualization*. The decline in *CAD(2D)* development with the increase in *3D Modeling* establishes that a once prominent category declines while development in another category increases and eventually overtakes the original category. This also occurs in *3D Visualization* in which there is an increase in development of software with a primary function associated with *3D Visualization* and a secondary function with *2D Visualization,* reversing early trends. If this trend continues, then *3D Visualization* may usurp *2D Visualization,* perhaps leaving *2D Visualization* as a secondary function. While currently theoretical in nature, as neither scenario has fully occurred, this potential trend is worth mentioning as this would indicate a significant change in direction within the industry.

## CONCLUSION

The SDGR displays developmental trends of software used in architecture. Several patterns emerged from its analysis. These patterns may be able to indicate trends within software development that relate to larger trends and the software's use within architecture. Of these the following patterns are considered significant.

- The segregation of the taxonomies and development within them is related to the software's industry of origination, primary functions, and its associated use. This impacts how software interacts with the taxonomies and how it is used within architecture.
- Plug-ins can be used to indicate the importance or strength of the relationship between taxonomies.
- Development of plug-ins or secondary functions can be used as a predictive pattern to indicate the importance of the associated function to architectural practice.
- Primary categories, when prevalent in a taxonomy, can be viewed as measures for the taxonomy and industry.
- The changeover of a primary category within a taxonomy may be indicative of changes in practice or direction within the industry.

The encoding of software in the SDGR has aided in the understanding of software development in the AEC industry, its relationship to AEC disciplines, and how this information may be used to relate the observed trends to developmental trends within the industry. The SDGR format and the discussed patterns, when overlaid with larger trends in the AEC industry, may be able to provide additional insight into the development of the architecture discipline. Future studies will evaluate how this information may be used in conjunction with information such as a timeline of major projects, delivery methods, and software adoption to understand the relationship between software and larger trends within the AEC industry.

## REFERENCES


Aksamija, Ajla. 2016. *Integrating Innocation in Architecture*. Chichester, West Sussex, UK: John Wiley & Sons.

Aouad, Ghassan, Song Wu, Angela Lee, and Timothy Onyenobi. 2012. *Computer Aided Design Guide For Architecture, Engineering and Construction*. New York, New York: Spon Press.

As, Imdat, and Daniel Schodek. 2008. *Dynamic Digital Representations in Architecture*. New York, New York: Francis and Taylor.

Frazer, John. 2016. "Parametric Computation: History and Future." *Architectural Design* 86 (2): 18–23.

Galofaro, Luca. 1999. *Digital Eisenman*. Basel, Switzerland: Birkhauser.

Grobman, Yasha. 2012. "The Various Dimensions of the Concept Of 'performance' in Architecture." In *Performalism: From and Performance in Digital Architecure*, edited by Eran Neuman and Yasha Grobman, 9–13. New York, New York: Routledge.

Kalay, Yehuda. 2004. *Architecture's New Media: Principles, Theories, and Methods of Computer-Aided Design*. Cambridge, MA: MIT Press.

Kvan, Tom. 2009. "Debating Opportunities: Learning Design through Different Structures." *Mixed Reality in Architecture, Design and Construction*, 227–36.