How I Stopped Worrying and Learned to Love AutoCAD

Mark J. Clayton[1]
[1] Texas A&M University

Abstract

The history of computing is expressed through AutoCAD as an accretion of ideas and inventions, each of which was a breakthrough in its time. Learning to use AutoCAD, or any CAD system, is augmented by an understanding of the historical context of its development. In contrast to a "deconstructivist" criticism of AutoCAD that avoids all historical context, this paper discusses the user interface of AutoCAD placed in its historical context by combining facts of history with personal reminiscences. The paper answers mysteries about AutoCAD such as "Why a black screen?", "Why LISP?", "Why a command line?", "Why layers, pens and line types?", and "Why 2D?" An understanding of context and history is a starting point for understanding, mastering, and improving software.

Mortality and the Recognition of History

It is a peculiar sensation to realize that one's own experiences and past have become history, particularly from the vantage point of merely middle age. In these times of rapid change, the old and familiar rapidly become the strange and forgotten. Computing has grown from non-existence to pervasive impact upon every aspect of human life in a time period of less than the lifetime of an individual. PC CAD (an acronym that may already be slipping into the obscuring mists of history), spans little more than a human generation, and in my case, neatly encompasses my professional life. When I was entering college, the personal computer was being invented. By the time I finished college, AutoCAD had reached the market. In my first jobs as a professional designer I became a CAD manager. I started some of the earliest courses in architectural computing. I pursued graduate studies to increase my knowledge and hone my skills at the cutting edge of architectural computing. Along the way, I learned to use AutoCAD, Microstation, VersaCAD, Dynaperspective, DataCAD, ProDraft, PowerDraw, MiniCAD, ArchiCAD, FastCAD, Arris, Sonata, Reflex, EasyCAD, ComputerVision, CATIA, and many other forgotten CAD systems. AutoCAD is still with us, and in fact dominates the contemporary architectural tools market. It has competed successfully in the race to innovation and adoption during that entire time. Under the hood and in the undercarriage is an entire history of computing. Examination of that history can give the reflective student, those who ask not only what and how, but also why and when, a way to reach understanding that can be the basis for progress.

This paper is presented as a counterpoint to an article prepared by Mahesh Sengala that "deconstructs" the user interface of AutoCAD (Senegala 2004). That article accuses the makers of AutoCAD as either consumed by ideological subversiveness or being "dim-witted" (305). He supports this idea by reference to characteristics of the AutoCAD user interface. Senegala suggests that subversive characteristics include the command line, the "black void" screen, the inadequate treatment of line weights, the preference for 2D, and the disorderly collection of icons, menus and keyboard commands. However, AutoCAD employs people who are certainly not dim-witted and are more pragmatic than subversive. There are reasons, some historical and some practical, behind all of the idiosyncrasies of AutoCAD. It is more profitable to use the idiosyncrasies to explore the world of ideas clustered under the rubric of architectural computing.

I suggest that the deconstructionist critique suffers from an impoverished concept map due to its insistence on disregarding evidence from outside the artifact in question. At the risk of oversimplification, I present a concept map in Figure 1 to summarize Senegala's argument. The self-referential approach is limiting and inadequate. Context is important. I will progressively develop a richer concept map that will lead the reader toward ever expanding circles of exploration. In the conclusion, I hope I can present convincingly that the "subversion" in AutoCAD is actually in the mainstream of architecture as a discipline, practice and profession. The view of architecture expressed in AutoCAD clashes not with the mainstream of the profession but with a limited and limiting view of architecture that might be called "the cult of the sketch."

Perhaps my approach is post-modernist rather than deconstructionist, as I will employ an attention to history. I attempt to ground the discussion in context and integrate an extensive range of ideas. I prefer to think of this paper as a documented oral history, personal memoir, or even a testimonial of a personal journey to enlightenment. Perhaps such an approach is inevitably more gentle, reflective, and compassionate than the pretended dispassion yet superciliousness that seems characteristic of deconstructivism.

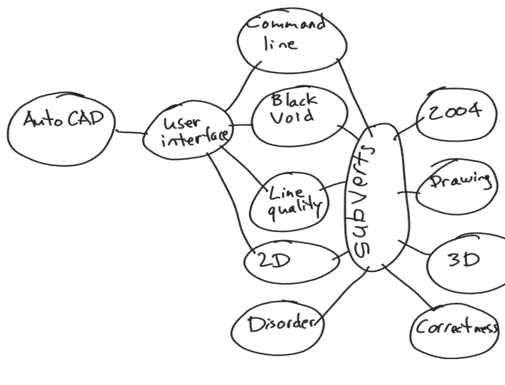


Figure 1. *Concept map of the view that AutoCAD subverts architectural traditions*

A chronological personal memoir would be a self-centered indulgence. Alternatively, I have organized this paper into five themes that combine Senegala's points of criticism with historical context and personal reminiscences. These themes are 1) computational design; 2) evolution; 3) overlay drafting; 4) 2 or 3D; and 5) disorder, information and process. The final section summarizes these themes to expound an empirical theory of architecture generated from the forty-year discourse on computing and architecture that has generated AutoCAD.

The Command Line and Computational Design

Any computer system presents strange, idiosyncratic, or cryptic ideas and languages to the new user. I have watched students struggle with MKDIR, grep, or dragging a floppy disk to a trashcan. With experience, confidence and familiarity increase, and the operations become obvious and comfortable. Although the command line in AutoCAD is much maligned as antithetical to a graphic sensibility, for an experienced AutoCAD user it is a component of a highly optimized, two handed user interface. The use of the space bar as equivalent to the "Enter" or "Return" key is a brilliant feature that eliminates handedness; AutoCAD is equally usable (or unusable) by left-handed people as right-handed people.

It is easy to mistake the command line as counterintuitive or anti-graphical. It is also easy to hide. However, it is also a persistent and beautiful reminder of an era when architecture was not graphical, and CAD was not a branch of computer graphics. My introduction to architectural computing as a formal subject of study comes from spending 1986 and 1987 at UCLA. Bill Mitchell, Lionel March, George Stiny, Murray Milne, Robin Liggett, and Jeff Hamer espoused a commitment to "computational design." Fellow students and I learned to program as a means to generate design from functional and aesthetic criteria. We programmed windows and facades that could be controlled in pattern and proportion by use of parameters (Mitchell, Liggett and Kvan 1987). We wrote analysis programs for space layout and technical analysis, including Solar 5, Opaque, and Climate Consultant (Moore, Milne and Geier-Wilson 1993). We devised shape grammars for Palladian Villas, Prairie Houses and numerous other building types (Mitchell 1990). Our work was merely a moment in a continuous dialogue about computing that was initiated at Cambridge in the late 1950s (Keller 2005). Before the invention of computer graphics, architectural computing explored optimization, space layout, proportion, pattern, and other topics. By the 1970's, computational design was producing large, complex buildings such as hospitals that were demonstrably superior to those designed with conventional methods (although aesthetically unacceptable to the general public) (Hoskins 1977). In truth, the dialogue is much older than computing and includes many of the preeminent architects in history, as thoroughly expounded by Mitchell (1990).

The AutoCAD command line is a reminder and acknowledgement of the contributions of that era. By expanding the command line one can scroll through an entire AutoCAD session to uncover the sequence and precise steps in creating a drawing. One can use that record to gain insight into design process or generalize a session into a repeatable automated sequence.

Through the command line, Auto LISP is always ready to spring into action to automate a repetitive task or apply reasoning to a design problem. Invented in 1958 as one of the first high-level programming languages, LISP is still the preferred language for much artificial intelligence investigation (Spencer 1997). For many years, Auto LISP was a common tool for architectural investigation and remains a quick and dirty way to manipulate data and graphics in AutoCAD. Auto LISP may be a decisive factor in the dominance of AutoCAD as it enabled the sales force to tailor AutoCAD for the needs of specific clients. It is puzzling why other entries onto the CAD market have neglected the inclusion of macro languages. A command line is an asset, not a liability.

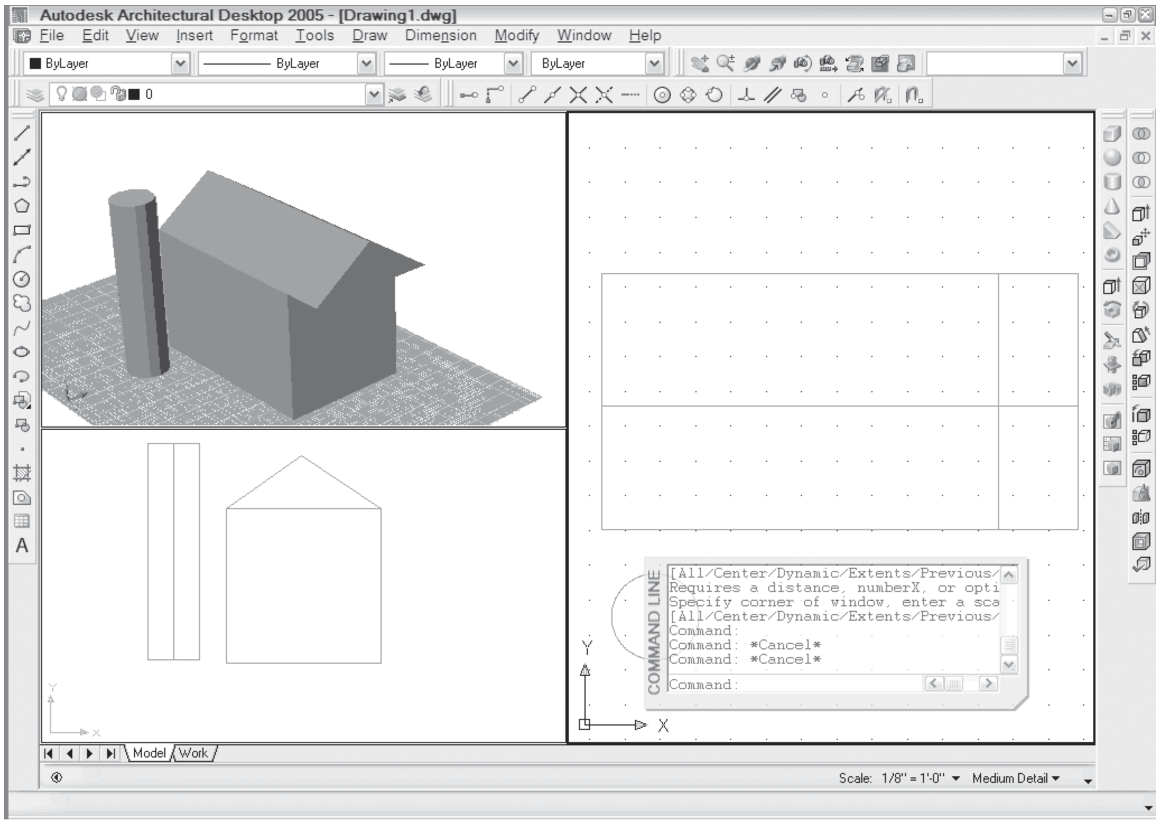
The Black Void and Evolution

When AutoCAD starts, the default screen is a black field with no indication of size, or direction. Senegala describes this "black void" as another subversion of an architectural sensibility. Of course, one can easily change preferences and defaults so that AutoCAD opens with a white background, appropriately sized grid defining the ground plane, and perspective view, as illustrated in figure 2. However, the black void has historical associations that help us remember where we were, where we are now, and where we may be going.

AutoCAD was announced as a software product in 1982, one year after the introduction of the IBM PC and two years before the Apple Macintosh (Spencer 1997). The innovation of a hard disk in a personal computer was still one year away, awaiting the release of the IBM PC-XT. It was a startling software "hack" to fit a full-featured CAD system onto a 360kb floppy disk to run in less than the 640kb limit of RAM enforced by MS-DOS. In that era, most computing was done in text mode; it took a special display and a special graphics circuit board to display graphics. A CGA card displayed 4 colors at a resolution lower than that offered by my cell phone. Word processors operated in text mode sans any font display or page layout capabilities. Screen space was very precious. I suspect that the prototypes of AutoCAD were written to allow the user to interact in text mode and then switch to graphics mode to view the results.

The question of how to interact with a computer, particularly for graphics, was wide open. In 1984, I was an intern-architect assigned largely to CAD drafting. The firm leased one workstation of the ProDraft system by Bausch and Lomb for upwards of $80,000 per year. The computer was dedicated to CAD and in fact could run no other software. It used a proprietary operating system. Input was through a keyboard and a digitizer tablet; the mouse had not yet been introduced to the popular market. There were no pull-down menus or on-screen icons.

In 1987 when I joined the faculty at Cal Poly, there had been steady progress in graphics software, but no earth-shaking changes. Our computer labs contained PCs and PC-XT's with a few PC-AT's for experimental purposes. The state of the art was AutoCAD, VersaCAD, and DataCAD, each of which employed screen menus for user interaction and a mouse as graphical input device. Figure 3 approximates the look and feel of AutoCAD from that era, as obtained from adjusting preferences in AutoCAD 2005.


Autodesk Architectural Desktop 2005 - [Drawing1.dwg]
File Edit View Insert Format Tools Draw Dimension Modify Window Help
ByLayer
ByLayer
ByLayer
ByLayer
0
COMMAND LINE
Y
X
Model
Work
Scale: 1/8" = 1'-0"
Medium Detail

```
[All/Center/Dynamic/Extents/Previous/
Requires a distance, numberX, or opti
Specify corner of window, enter a sca
[All/Center/Dynamic/Extents/Previous/
Command:
Command: *Cancel*
Command: *Cancel*
Command:
```

Figure 2. *A 3D modeling setup in AutoCAD*

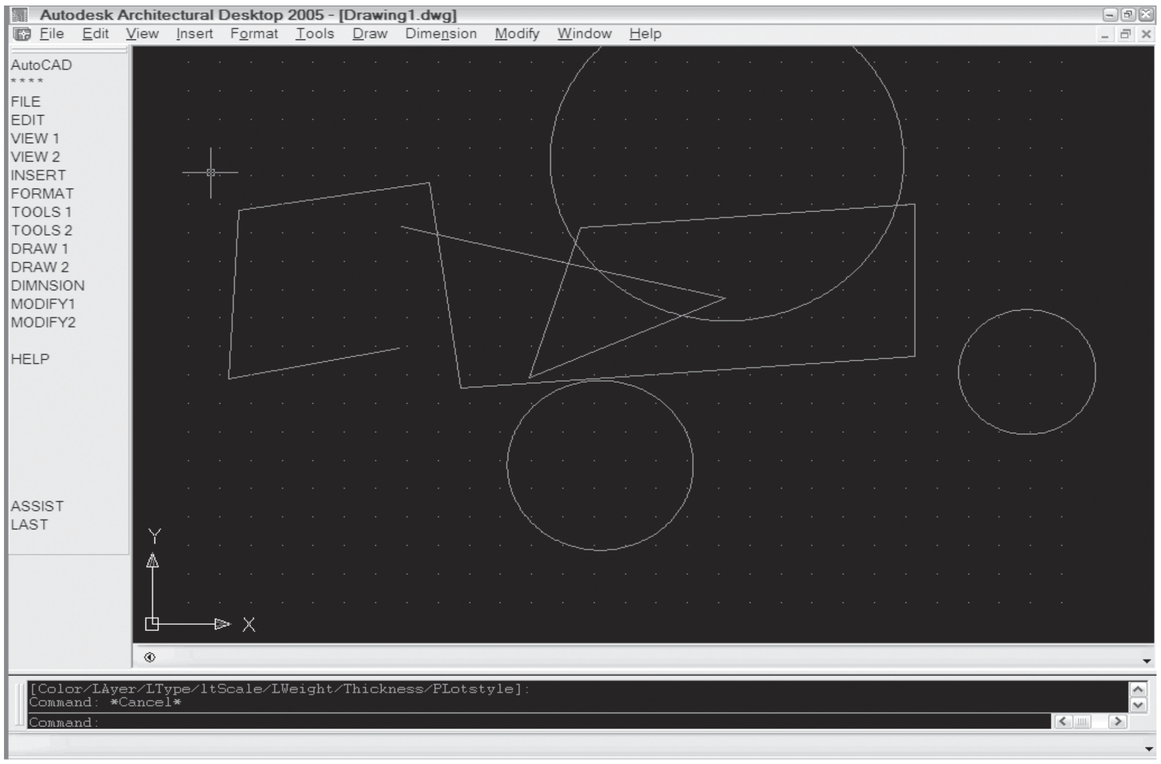
The black void screen, to one who knows the history, is an expression of economy and minimalism. It is a reminder of the cleverness and discipline that was required to achieve PC CAD. AutoCAD out of the box is a neutral tool that provides a tabula rasa for the exploration of design ideas, but if you don't like it, then change it. An appreciation of the immense computing power of current computers in comparison to those of ten or twenty years ago may temper the profligate wastage of such power due to shoddy modeling. There is value in educators focusing upon how to build compact, efficient computer models that can fit into modest computer resources.

Line Weights and Overlay Drafting

The ascription of subversiveness to the implementation of line weights in AutoCAD can only be justified by ignoring the purpose and origins of the software. The graphic language employed in AutoCAD is not that of fine art drawing and sketching, but that of mechanical drawing. In mechanical drawing, the quality of lines is symbolic, not aesthetic or expressive. AutoCAD derives from a long graphic tradition that prizes clarity and meaning rather than appearance. Once again, the historical perspective helps to clarify one's understanding and perhaps increase one's ability to use the tool.

In 1978, drafting was an overwhelming part of architectural practice and education. As a student at Tulane, I had an enormous advantage in that I could not only draw but I could draft. My lettering was terrible, but so was that of everyone else. I understood perspective, something that set me apart from 95% of my classmates. Very few students ever attempted a perspective, and the instructors generally encouraged us to work in plan and section. The long nights of work in studio were spent in producing the final drawings for presentations, which were mostly drafted using ink on vellum and consisted of 1/8" = 1' or ¼" = 1' plans, sections and elevations.

The faculty had modest drafting skills; they could draw, but they knew nothing about emerging technology for documentation that was poised to


Autodesk Architectural Desktop 2005 - [Drawing1.dwg]
File Edit View Insert Format Tools Draw Dimension Modify Window Help
AutoCAD
* * * *
FILE
EDIT
VIEW 1
VIEW 2
INSERT
FORMAT
TOOLS 1
TOOLS 2
DRAW 1
DRAW 2
DIMNSION
MODIFY1
MODIFY2
HELP
ASSIST
LAST
Y
X

```
[Color/LAyer/LType/ltScale/LWeight/Thickness/PLotstyle]:
Command: *Cancel*
Command:
```

Figure 3. *Simulation of AutoCAD circa 1987 using settings in AutoCAD 2005*

revolutionize practice. Using the new tools, the time needed to draft a project could be reduced by an order of magnitude. This technology was not the computer; it was overlay drafting. Widespread in the manufacturing industry, the technology was adopted slowly by the architectural professional. The accidents of my personal experience introduced me to the technology that became the foundation of the CAD metaphor that we know today.

While working in Washington, DC in 1980, draftsmen introduced me to the tricks of their craft, such as lead holders, micro-leads, rapidographs, parallel rules, plastic templates, drafting machines, triangles, and French curves. I learned to draw on both sides of a sheet to make alterations easier. I became proficient at using a Leroy set for lettering. Through friendships with the blue-collar workers in the printing companies, I received tours and demonstrations of the equipment that gave them pride. They showed me vacuum press blueprint machines and full-size photography rooms. They showed me how to make sepia "backgrounds", photographic negatives, and composites out of multiple sheets. A photocopy machine (a relatively new piece of office equipment that many architects believed was an unneeded luxury) could be used to copy repetitive drawings, such as standard details, entourage, and even plumbing symbols, onto transparent adhesive film (sticky back paper). These could be adhered to a drawing, eliminating hours of work. Dry transfer lettering increased the legibility of text, substituting for a skill that distinguished an experienced draftsman from a newcomer. Patterns pre-printed onto adhesive film enabled large areas of hatching to be created quickly and precisely. Even colors and tones could be easily added as transfers.

I bought a pin bar for registering multiple sheets of mylar to support overlay drafting, which I used at home on competitions that I entered as a student. I read books on advanced drawing production, such as those by Fred Stitt (1984). In retrospect, I realize that I acquired an advanced knowledge of drawing production that was far beyond what I could have learned from professors and schools. Overlay drafting was the cutting edge of architectural production of the day.

When software developers invented CAD systems, they mimicked the state of the art of drafting in their day. Overlay drafting became the model for CAD systems. AutoCAD employs a typical overlay drafting metaphor, as does nearly every CAD system on the market. The concept of layers became the dominant and ubiquitous way of organizing graphic information. The plastic tracing templates also received a digital counterpart, variously called symbols, blocks, templates, or cells in different CAD systems.

The overlay metaphor for digital representation of drafting has proven effective in comparison to manual drafting. It is nevertheless a clumsy way to represent a building. Buildings are not made of blocks, layers, line styles, and line weights, nor Xrefs, paper space, or sheets. The traditional drafting representation is itself a metaphor for a building. Nearly all CAD systems, AutoCAD included, enforce a metaphor of a metaphor onto the hapless architect.

Like all metaphors, the idea of overlay drafting breaks down when translated for the computer. Line types and line thickness are tough problems in a world of infinite zoom, and the solution in AutoCAD does not seem ideal. Pen settings in AutoCAD are inevitably annoying, but are understandable in the context of pen plotters. In the 1980's, the line weight depended not on a software setting but upon which pen was in which location in the pen carriage of the plotter. The distinction between paper space and model space helps to correct the metaphor by applying more tangible aspects of the metaphor, such as line weight, color, and visibility, in paper space. Model space can then be focused upon representing the building itself, where line weight, line style, or color have no real meaning beyond their symbolic importance.

Autodesk has partially replaced the overlay metaphor in Architectural Desktop 2005. In the new metaphor, an element is a conceptual part of a building that may be repeated. A construct is a conceptual part of a building that is not repeated and is made largely of elements. Various views of constructs can be arranged into sheets, sheet sets, and a project. The metaphor is based on the latest thinking regarding how to put together a large set of drawings to describe a large building. By implementing the metaphor with external references (Xrefs), Architectural Desktop can manage hundreds of sheets and thousands of drawings while maintaining adequate performance on commodity hardware. Building Information Modelers promise to substitute a new metaphor that is focused upon the physical components of a building.

Although the overlay drafting metaphor is ubiquitous, it is neither inevitable nor unique. For example, a cognitive metaphor has been suggested as more supportive of the design process (Clayton et al 1994). Rather than grouping objects by layers, the graphic objects are identified within interpretations as features that have significance in various tools for evaluating the performance.

2 or 3D

Senegala suggests that AutoCAD imprisons designers in a two-dimensional flatland. I hear and read this criticism frequently, but it is incongruent with the actual software. Few people seem to realize it, but AutoCAD incorporates powerful surface modelers and solid modelers based on geometry libraries that are among the best available in the software industry. Its object-oriented solid modeler provides for Boolean operations on solids as well as extrusions along arbitrary paths, shell commands, and face-editing. The built in renderer performs very well, providing both ray tracing and shadow mapping, and extensive texture mapping, background compositing, and entourage. The techniques are well-explained in a widely used textbook for architectural computing (Kolarevic 1998).

A historical perspective can explain the widespread ignorance about the 3D capabilities in AutoCAD. Old impressions gained before the addition of more powerful functions seem to persist in spite of additions to the feature set. One's image of AutoCAD seems colored by the version that was current at the time of first impressions. The early focus of the software on 2D drafting is easy to understand in the context of the demand in industry for overlay drafting and the limitations of the IBM PC hardware.

AutoCAD evolved at a pace with the rest of the industry with respect to 3D. For example, in the late 1980s, VersaCAD on the Macintosh, provided only surfaces extruded parallel to the Z-axis. AutoCAD still provides this version of so-called 2 1/2 D modeler through manipulation of the thickness property. Surface modeling is also available in AutoCAD using the 3Dface command. Ruled surfaces, surfaces of revolution, surface meshes, and Euclidean primitives are all provided. Auto LISP is available for automating the creation of meshes. It is relatively simple to generate a table of vertices in Excel as a table and execute it in AutoCAD. The Advanced Modeling Extension (AME) was introduced in the early 1990s as a pioneering Constructive Solid Geometry system that was far more advanced than the surface modelers available in competitor's products. With Release 13, solid modeling was fully integrated into AutoCAD.

More recent versions have provided ever more powerful tools for editing and viewing 3D models, as illustrated in figure 2. A right click on the AutoCAD screen pops up a menu that includes the 3D Orbit command, from which one can rapidly and intuitively set up perspective views. For even more advanced 3D modeling, Autodesk provides tight integration with 3D Studio Viz. AutoCAD is clearly an extensive environment for 3D modeling. It is particularly valuable for teaching 3D modeling by virtue of its tools for so many different 3D paradigms. A student can compare 2 ½ D to mesh modelers and constructive solid geometry.

Disorder, Information and Process

One final criticism of AutoCAD deserves attention: the user interface is disorderly and lacks a coherent model for a design process. This criticism betrays a profound lack of understanding regarding the stated market for Autodesk products. In the 1980s when AutoCAD s hegemony was surprising, the conventional wisdom was that the company had hit on a fortuitous formula for selling CAD software to architects. AutoCAD was easy for dealers to customize. Thus they could sell block libraries, Auto Lisp routines, menu customizations and training to their clients. Architects wanted and needed the hand-holding provided by a registered AutoCAD dealer and the dealer could make profits far beyond the margin on the retail price by selling services. The smart dealers sold AutoCAD and the smart architects found a smart dealer.

AutoCAD has always been a CAD engine rather than a finished shrink-wrapped, turn-key product. The purchaser is supposed to customize it for the needs of a particular firm. It is now and in the past also was irrelevant if the user interface was unappealing to a particular user. If you like the menu structure, keep it. It you don't like it, change it. The software can adapt to any process.

AutoLISP is not the end of AutoCAD customization tool development. In the early 1990s, the AutoCAD Runtime Extension (ARX) libraries were devised to allow a developer to write programs in C, compile them with any typical C compiler, and link them into a running copy of AutoCAD. This was and is an amazing trick that enabled developers to conceive and implement extensions to AutoCAD virtually without limits in terms of scope and speed of execution. An example of an ARX tool created in a research lab in the mid 1990s is the Semantic Modeling Extension, which tied together expert systems, automated consultants, and AutoCAD on multiple, heterogeneous machines across the Internet (Clayton et al 1999)

Once Autodesk focused exclusively on the Microsoft platform, the company moved quickly to incorporate the Microsoft Component Object Model (COM), exposing its entire class hierarchy to developers for manipulation from within any COM compliant application. A conversant programmer can run AutoCAD commands from within Excel or call Excel routines from within AutoCAD.

COM programming is not for the novice, but other features in AutoCAD provide many levels of integration with data and software. Block attributes and extraction enable quick and accurate quantity take-offs and furniture tracking. Data linking provides dynamic, real time communication with database systems. The hyperlink property is available to link any entity with a URL. Blocks can be provided with behavior and drawings located on the Web dropped into a drawing. AutoCAD continues to provide capabilities for process modeling and data integration that are easy to use and accessible, although rarely used by the average user.

Of course, with the Desktop series, Autodesk has provided CAD software tailored to specific markets. Architectural Desktop 2005 provides a carefully designed process for building design. Civil Desktop and Land Desktop tailor AutoCAD for civil engineering and landscape architecture. Autodesk Map 3D turns AutoCAD into a powerful GIS system. No competitor vendor, with the possible exception of Bentley, even attempts to address the design process of architecture and other disciplines that are responsible for the built environment.

Miscellaneous Rebuttals

Several other criticisms are directed at AutoCAD in Senegala's article. He suggests that a sketching tool should have a sense of ambiguity and should support multiple layers of undo and redo. AutoCAD is not intended to be a sketching tool, but is quite serviceable for exploratory 2D and 3D modeling. A user can be sloppy with drawing and then "grip edit" back to a grid or other snaps. One can edit the vertices of a polyline one by one to correct a sloppy line. If ambiguity is important to the user, then the latest version can scribble on one's drawings automatically to make them look soft and ambiguous. Photoshop can be used to smudge one's renderings to convey lack of commitment. With regards to undo and redo commands, AutoCAD has long supported powerful tools for settings points in a session, backtracking through the session, and then moving forward again.

Conclusions

By virtue of its 23 year history and dedication to preserve old features, AutoCAD is a rich document of the history of CAD. For one who can read AutoCAD, the software offers reminders of the best ideas from the past and explanations for the current state of architectural computing. From a historical perspective, the user interface of AutoCAD reveals a history of computational design and computer graphics. From the perspective of architectural practice, the argument is compelling that AutoCAD is an excellent tool that satisfies at a high level more needs than most competing products. A concept map of this discussion is provided in figure 4.

Deconstruction of Senegala's article reveals a hidden assumption and bias in favor of a view that drawing is indistinguishable from design, a view that one might term "the cult of sketching." By focusing on the dialectic between the AutoCAD user interface and hand sketching, he privileges a myopic focus on hand sketching. While this "stance" is very common among those in the academic community, from an objective standpoint, sketching is not what architects do. Architects collaborate, produce construction documents, manage business, design details, research materials, and act in many other ways. In a typical firm, a few architects out of many sketch and visualize grandiose schemes. Viewed from within the cult of sketching, the AutoCAD user interface may appear ludicrous. However, from a more complete view of architectural practice the AutoCAD user interface is highly optimized for production, collaboration, and information management.

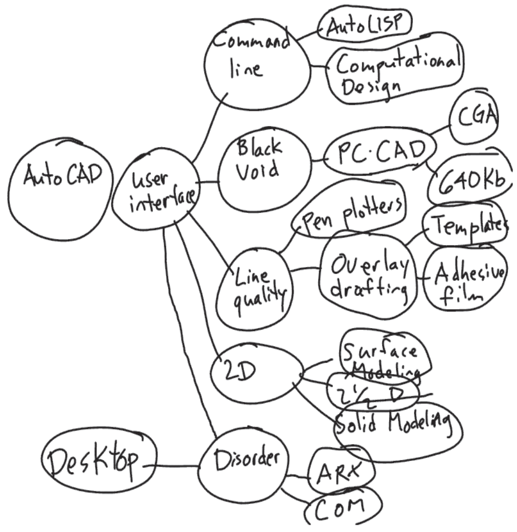


Figure 4. *Redrawn concept map critiquing the user interface of AutoCAD*

The tradition of computational design research and the critical importance of drafting production are two truths of architectural computing that belie the "cult of the sketch," Because it is aligned with the mainstream of architecture discipline and practice, AutoCAD is not subversive. However, it does throw down a gauntlet to those who would privilege the graphic image as the only expression of architecture.

My experiences have guided me to recognize that architecture is a diverse and challenging profession that engages one's intellect in mathematics, engineering, writing, management, and personal relationships as well as drawing. CAD has for me always meant computer-aided design in all of the myriad aspects of design. It has never meant computer-aided drafting or computer-aided drawing. AutoCAD puts few constraints upon the study and practice of architecture and is thus an impressive computer-aided design tool. In all its elderly crankiness, it is a lovable software package for those who have "been there, done that."

References

Clayton, M. J., Kunz, J. C., Fischer, M. A., and Teicholz, P. (1994). First drawings, then semantics. In *Reconnecting: ACADIA 94*. Association for Computer Aided Design in Architecture.

Clayton, M. J., Teicholz, P., Fischer, M., and Kunz, J.. (1999). Virtual components consisting of form, function and behavior. *Automation in Construction* (8) 351-367.

Hoskins, E. M. (1977) The OXSYS system. In J. S. Gero (Ed.) *Computer Applications in Architecture*, (pp. 343 – 391). Essex, England: Applied Science Publishers, Ltd.

Keller, S. (2005). 10283EFE0F02 or the Seagram Building. R. Hejduk and H. van Oudenallen, (Eds.), *The Art of Architecture/The Science of Architecture*. Washington, DC: Association of Collegiate Schools of Architecture Press.

Kolarevic, B. (1998). *Architectural modeling and rendering with AutoCAD R13 and R14*. New York: Wiley.

Mitchell, W. J. (1990). *The Logic of Architecture: Design, Computation and Cognition*. Cambridge, MA: The MIT Press.

Mitchell, W. J., Liggett, R. S., and Kvan, T. (1987). *The Art of Computer Graphics Programming: A Structured Introduction for Architects and Designers*. New York, NY: Van Nostrand Reinhold.

Moore, G. T., Milne, M., and Geier-Wilson, R. (1993). Architectural research. *Progressive Architecture* (74[8]), 83-85.

Senegala, M. (2004). Deconstructing the software interface: a critical close reading of AutoCAD, *International Journal of Architectural Computing* (3), 299-314.

Spencer, D. D. (1997). *The timetable of computers*. Ormond Beach, FL: Camelot Publishing Company.

Stitt, F. (1984). *Systems Graphics*. New York: McGraw-Hill Book Company