The Level of Knowledge of CAD Objects within the Building Information Model

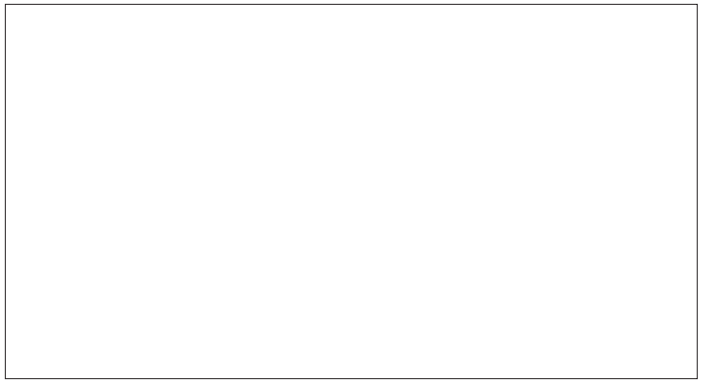
Magdy Ibrahim, Robert Krawczyk
Illinois Institute of Technology

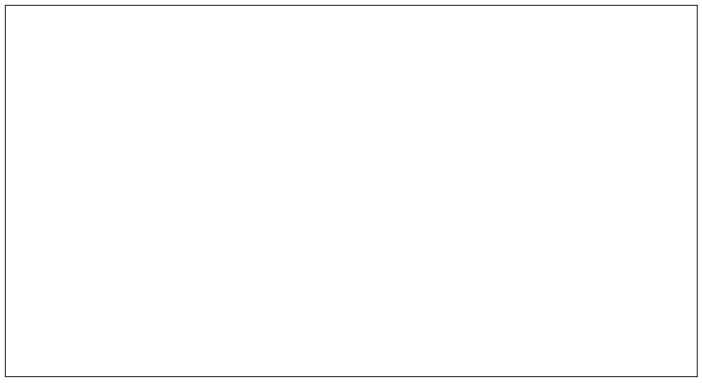
Abstract

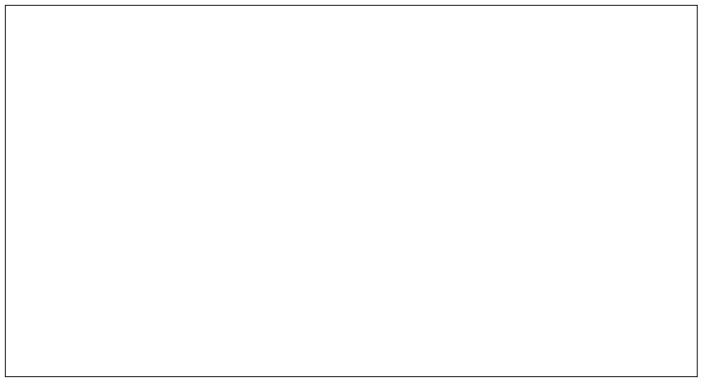
The first generation of CAD software depended on entity objects that were manipulated and interpreted by the user as meaningful graphics symbols. These entities only represented the geometrical properties of the architectural elements. With the present emerging generation of CAD systems, a new concept shifts a drawing-based model into a Building Information Model with the potential of modeling true architectural objects. Theoretically, these CAD objects will provide all related data to the designer describing the geometry, as well as any related data associated with how the object is actually used.

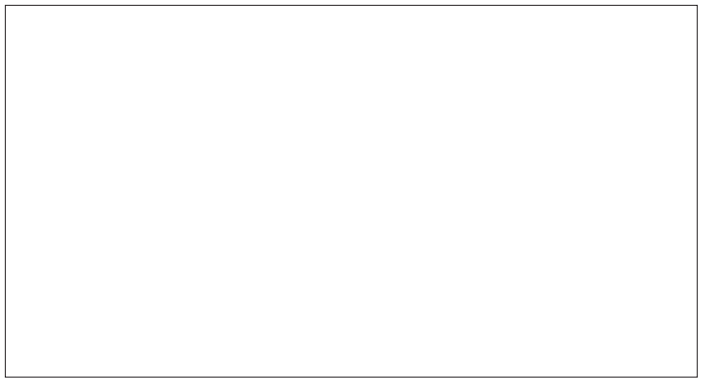
The knowledge required to support an object should have structure to it. Different levels of knowledge need to be included, such as the geometrical information, which should be flexible enough to accommodate any type of shape and modification while keeping the object's integrity as a unit and maintaining its relations to other objects.

The CAD object concept, as remarkable as it is, might also have potential problems. It has some implications over the design process, as well as the architectural profession itself.









The Level of Knowledge of CAD Objects within the Building Information Model

Magdy Ibrahim, Robert Krawczyk
Illinois Institute of Technology

The New CAD Concept: BIM

BIM stands for Building Information Model. In fact, the idea of BIM-based CAD is not new: it has always been foreseen as the ideal way to represent buildings digitally, but it has never been mainstream for commercial products until recently, mainly due to the increased capacity of personal computers. Graphisoft ArchiCAD Virtual Building concept may have been the first commercially available package that utilized the building model; now more and more CAD software is being built around this new concept. Object oriented programming is not new. All applications written in C++ are object-oriented, but it took the CAD industry some time to apply this software concept to the building elements themselves.

The building industry has traditionally communicated building construction information through drawings with notes and specifications. CAD technology automated that process, and object-oriented CAD extended the idea of adding information to graphics. The result of earlier manual drafting and CAD systems were identical, creating graphic abstractions of the intended building design. Intended for building graphics and generating two-dimensional printed/plotted drawings, these systems were capable of handling and managing information about a building only on a limited basis. Other industries, such as manufacturing, have realized great benefit from non-graphical, parametric information technology tools. The BIM-based generation of CAD systems, designed with current technology, is required to fully realize the benefits of object-oriented CAD. This next generation of information-centric software provides building information modeling instead of building graphic modeling.

Building information modeling operates on digital databases. By storing and managing building information as databases, BIM systems can capture, manage, and present data in ways that are appropriate and customary for a particular designer, contractor, vendor or client. Such applications start with capturing and managing information about the building, and then present that information back as conventional drawings view or in any other appropriate way such as tables or perspective views, and make it available for use and reuse at every phase in the project (AutoDesk 2002).

With the arrival of the BIM-based CAD, came new concept of objects.These objects are not only programming objects, but they have specific meaning to the architect. They have an equivalent physical meaning to real world objects, and provide an abstract computer representation of the physical world that is convenient for architects (Ruppel, Meissner, and Bernd, M, 1993). A wall as an object in the CAD system represents an actual wall in the physical world, and a door as an object represents a real door. With objects, all the standard object-oriented programming concepts apply. Objects have properties, methods, and events. The advantage of the object model is that it allows for the extension of the properties or attributes, not to be confused with the block attributes as found in AutoCAD.

Entities vs. Objects

An *object* from a computer science point of view is an independent procedure that contains both the instructions and data to perform some task, and the programming code necessary to handle various messages that it may receive (Morris 1999).

Both industry and academia have devoted countless hours of research and development to the problem of describing geometry digitally so it can be stored, presented and manipulated on a computer and a plotter. The “geometry engines” resulting from these efforts were, and remain, the core technology in today’s products (Revit White Paper 1998).

AutoCAD is an example of a C++ written, object-oriented program which used a general concept of objects to create the “drafting elements” or “drawing primitives” such as lines and arcs. This is where the confusion begins.

While AutoCAD itself is an object-oriented program, the objects it provided were only graphical objects or “entities.” Although such objects had all the concepts of programming objects, they were used mainly to draw a representation of well-understood drawings of highly symbolic information about the building. The architect must interpret the meanings of what has been drawn, in the exact same way as with physical drawings. Using the same drawing legacy, a replica of what could have been drawn by hand was created using the computer as a drafting system. Even the creation of a symbols library, blocks in AutoCAD, is very dependant on the previous knowledge of the symbols used by the profession and to some extent equals the use of drafting templates. The architect could have named the symbol any name and still can be able to use it the correct way, because the interpretation relies on how the designer uses it.

These drawing entities only include geometrical aspects of the real objects that they represent, and never had knowledge of what they are, or how to behave or interact with each other. With the exception of Blocks with Attributes and their equivalent in other CAD packages, which to some extent had included more information about themselves, all entities were just a collection of basic drawing primitives.

With the BIM generation of CAD systems, embedded information can describe the geometry, as well as materials, specifications, code requirements, assembly procedures, prices, manufacturers, vendors and any other related data associated with how the object is actually used. Mechanical CAD already has many of these features. A door as a smart object that understands its relationship to a wall and reacts accordingly should be a great help to the designer. The potential of using CAD smart objects is very appealing in the production phase of a project. Although the architect is not obliged to give full information about the object he is using, that object has blank attributes waiting for such input. As a result, in the schematic design phase an object could even be represented symbolically to the architect, a wall could be a line, and a door could be two simple lines indicating the door opening. As the architect advances to the design phases, these semi-defined objects should become better defined and takes more decisions about the building. Adding information will always be the architect's responsibility.

In the production phase, objects are in full throttle, as generating the bill of materials is a matter of a button click, and consequently the cost estimation is easily achievable. Any changes done to the building database will simultaneously be reflected in the entire set of documents the architect is responsible for: plans, elevations, sections, schedules, and bill of materials.

For a design firm utilizing BIM technology this means savings in time and resources needed to coordinate changes. Even with advanced current CAD methods, like external references and writing routines to automate drawing production, this would have not been achievable with the ease and accuracy a BIM-based CAD could deliver.

This can lead to a problem where the customization of CAD packages may not be possible. Technical support teams in most firms now can easily automate repetitive tasks, and develop even more intuitive tools to help in their production. With the new breed of CAD that depends on smart objects, the customization capabilities of the older and simpler entity-based systems might be in question. It can be foreseen that the same need for customization will take place no matter how sophisticated the software becomes, but the inherent complexity in the new systems may make it more difficult.

To be Smart, Objects Must Have Knowledge

Constraints and parametric variation, which BIM CAD is built around, have been active areas of investigation in computer-aided architectural design for over fifteen years (Eggink, Gross, and Do 2001). Parametric design relies on the ability to change an object's properties numerically without the need to redraw it. The CAD system takes care of the redrawing part of the parametric object, but the parameters themselves must be previously assigned to the object.

A "door" object has a finite set of parameters that dictate its shape. The coding of the door object has to include these parameters, and this requires previous knowledge of the parameters involved in the creation of a door (Figure 1). The very basic parameters of a door object are the geometrical

Parameter	Value
Thickness	50.8
Assembly Code	C1020
Door Material	Door - Panel
Frame Material	Door - Frame
Height	2032
Rail Width-Center	101.6
Top Panel	914.4
Trim Projection Ext	25.4
Trim Projection Int	25.4
Trim Width	76.2
Wall Closure	By host
Width	1727.2
Model	
Manufacturer	
Type Comments	
URL	
Description	
Assembly Description	Interior Doors
Type Mark	24
Rough Width	
Rough Height	
Construction Type	
Fire Rating	

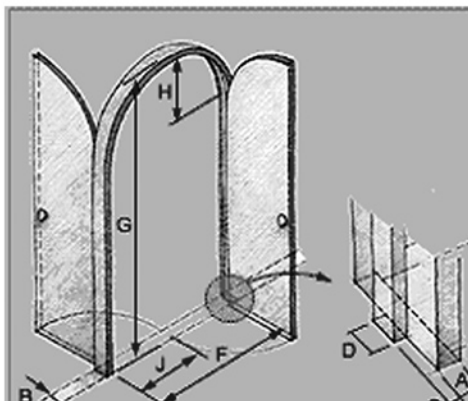


Figure 1. A "Door" object as it appears in the properties dialogue box in AutoDesk Revit 5.0 (left) and the different parameters connected to a "Door" object as it appears in AutoDesk Architectural Desktop 3.0 (right).

information, such as width, height, sill-height and frame dimensions. Parameters are always a predefined list of properties that the user has to select from, or abide by their rules for the manipulation or creation of a new object.

The necessary information for the set of parameters of a door object makes up architectural knowledge. The object will not be useful without the knowledge used to build it. But the problem is that the object is as good as the knowledge behind it.

Adding more information to that object becomes a question because the way the code is written may make it hard for an average user to manipulate. The object itself may become a prisoner of the predefined parameters, giving little chance for changing its character unless an entirely new object is defined. The knowledge required to support an object should have a certain structure to it, as the concept of the smartness will depend on how complete and how smartly the embedded information reacts to other objects. Different levels of knowledge need to be included, such as:

1) Geometrical information:
   It should be flexible enough to accommodate any type of shape and modification to it while keeping the integrity of the object as a unit and maintaining its relations to other objects.
2) Non-geometrical information:
   Materials, codes and assembly requirements should also feed back to the geometry of the shape by setting the limits of the possibilities the form may take.

Online Knowledge

The architectural profession relies on vendors and manufacturers' catalogs for specifications. For example, doors are rarely a custom design. Architects depend on a pre-designed unit selected from catalogs, which include the knowledge needed for the creation of the object inside the CAD system. For example, if an object could extract the needed information from a web page catalog written in XML, and change its parameters accordingly, then all the knowledge at the manufacturer side has been transferred to the object provided by that that object is coded to handle that kind of information. This will be accomplished if the main players in the industry agree on making it happen. In the production phase of a project, correctly and automatically adding all the specification related to an item will be a great benefit to the architect.

Some issues can be concluded here:

1) Objects should allow the designers the flexibility to design what they think without being limited by what is implemented.
2) Objects are like catalog items: they must be well connected with real vendors' catalogs to access their necessary parameters.
3) Vendors must agree on a standard for specifying parameters that is compatible with different CAD packages.
4) Objects should be flexible enough to adapt with the accumulation of information attached to them, throughout the design process.

How will this Change the Profession?

As remarkable as it is, the current development of the smart CAD concept might have a set of negative implications.

The Master Apprentice Relationship and CAD

CAD objects enable embedded information to be the primary reference for an architect, placing less emphasis on the master apprentice model that has developed in architectural practice. Young architects will not need technical support from their seniors as long as the computer provides this feed back. Will this lead to a new kind of draftsperson, who might not require much help and will complete a task with relatively fewer mistakes? How will this affect the development of problem solving skills, the professional education of an architect, and the master apprentice relationship in the workplace?

(Eggink, Gross, and Do 2001) presented a model for smart objects that would change its parameters depending on its boundary conditions; a beam-as-an-object should set its depth according to the span it covers, and change accordingly as the span changes. Another example, a door object should automatically set its fire rating according to the room where it belongs, depending on its knowledge of the room's function and code requirements. Theoretically, this can be achieved, and should enhance quality control in drawings. Although such intelligent behavior has not yet been implemented in current CAD programs, it is this kind of development that will be expected in the future development.

Evolution of Objects

How are objects represented during different phases of the design process? From the schematic design to design development, objects should have empty properties that might be added while the design is being developed. Can these properties/attributes be added dynamically as needed? It is possible to make objects dynamic enough to expand their knowledge as needed, like containers which will keep filling up with information during the different design phases. This should not be limited to a previously determined set of parameters, but to the possibility of easily adding more parameters as needed.The architect should be able to add more specifications to the object as parameters not just as textual description. There is no indication that current developers have addressed this issue.

Innovation and Creativity

Should smart objects allow for nonstandard uses? The smart object concept may not allow the architect to combine non-compatible components. How might this affect innovation? Pre-defined rectangular shape of a door opening from an object is normal, but what about the other door shapes? Unless supported by the vendor the current software does not allow architects to create their own non-orthodox objects.

What will the support group within a design firm need to know about software to change its parameter to support more nontraditional concepts? The architectural student might need to be trained differently to cope with this kind of change in the tools, understanding the potentials and limitations of the tools, and above all having a good understanding of the theories behind these tools. As future architects and IT support those students will constitute the users for such systems. The worst case changes will not be allowed.

Dependability on The Software Industry

If firms cannot easily add new objects, must they wait until the next version of the package comes to market? The obvious answer is yes. This is what is happening now, waiting for a curtain-wall tool in AutoDesk ADT, while other companies race to provide the same feature remain competitive. This kind of healthy competition benefits end users, but they must wait for a tool to be implemented. Consequently, architects will invest more money to obtain new features through upgrading.These systems might eventually become comprehensive enough for most users, but there will always be some special needs not foreseen.

Conclusion

It is very important to define the potentials of the building information model, and the implementation of objects in architectural CAD. The benefits of such systems will depend on better understanding of architects' needs in the office. Paying attention to the potential problems might make these systems more successful in the future.

References

Alshawia, M., and J. Underwood. (1996) Applying object-oriented analysis to the integration of design and construction. *Automation in construction*, 5(2):105-121.

AutoDesk, white paper, Building Information Modeling, *Autodesk building industry solutions(2002)* San Rafael, CA: Autodesk,Inc.

Autodesk Revit. White paper. The parametric building modeler: Answers to technical questions.

Beucke, K., and D.Rang Lack, (1993). Computing with objects: What does industry hope to gain from it. *Computing in civil and building engineering: Proceedings of the Fifth International Conference (V-ICCCBE)*, Louis F. Cohn. Anaheim, California, the American Society of Civil Engineers.

Eggink, D., M.D. Gross, and E. Do. Smart objects: Constraints and behaviors in a 3D design environment. *Architectural information management: 19th eCAADe conference proceedings*, Hannu Penttil,460-65 Helsinki,Finland,Helsinki University of Technology (HUT), 29-31 August 2001.

Fischer, T., M. Burry, and R. Woodbury. (2000) Object-oriented modeling using XML In *Computer-aided-architectural and educational Cad. CAADRIA 2000, Proceedings of the Fifth Conference on Computer Aided Architectural Design Research in Asia*,145-155, Singapore, 18-19 May.

Henkemans, D. and M. Lee. (2001). *C++ programming for the absolute beginner*. Premier Press.

Rozmanith M. Product management, autodesk revit. White paper. The parametric building modeler: Answers to technical questions.

Revit. White Paper. The Aec technology platform question.

Revit. White Paper. An introduction to revit.

Reymendt, J. and J. Wörner. (1993). Object-oriented modeling for concrete structures. *Computing in civil and building engineering: Proceedings of the Fifth International Conference (V-ICCCBE)*, Louis F. Cohn. Anaheim, California, The American Society of Civil Engineers. June.

Rudolph, D. (1999). *Mastering AutoCAD 2000 objects*. Alameda, Calif: Sybex, December.

Stephen M., *Object-oriented programming for Windows 95 and NT.* Digital Press, 1999.

Ruppel U., U. Meissner, and M.Bernd.(1993).Object-oriented data exchange for the integration of design processes in structural engineering," *Computing in civil and building engineering: Proceedings of the fifth international conference (V-ICCCBE)*, Louis F. Cohn. Anaheim, California, The American Society of Civil Engineers.

Weisfeld, M. A. (2000). *The object-oriented thought process*. Indianapolis, Ind.