chapter 1

Computer-Aided Design and Building Information Modeling

This chapter introduces the distinction between design automation and design intelligence in computational architecture. It presents the progress of computational architecture as an evolution from design automation to design intelligence. It discusses Computer Aided Design (CAD) and Building Information Modeling (BIM) as design automation paradigms providing a context for the subsequent development of Computational Design and Parametric Design. It describes, briefly, the historical development and characteristics of the CAD paradigm. It also describes the BIM paradigm in terms of its origins, the rationale for BIM and the benefits of BIM.

The use of computation in architectural practice has been an evolution from automation toward intelligence. While these are related concepts, we can still make an important distinction between them (Figure 1.1). Automation refers to using computers to perform repetitive tasks. For example, in creating a hatch pattern, the computer can draw hundreds or thousands of small squares very rapidly, while a designer would take a long time to perform the same task manually. Not only does it take a long time to perform, but there is also a threshold of repetition beyond which a human designer will find the task intolerable.

Intelligence, on the other hand, refers to a computer that has agency in the design process. The computer is not simply repeating a clearly specified mechanical process; it creates design elements and establishes design relationships without explicit user instructions. There is a clear sense of emergence. The computer infers geometry and geometric relationships from design requirements and constraints then actualizes them using design algorithms.

This drift toward design intelligence, while seemingly inevitable, has been controversial to some design practitioners. These designers question intelligent parametric design as a legitimate design methodology. Can architecture really be created by machines? Is there something specifically and unavoidably anthropic about architectural design? Do you need to be capable of human experiences to design the spaces in which human beings exist?

A fortiori, it can be argued that computers can now purport to paint pictures and write poems. However, no computer has created a widely acknowledged masterpiece of art or literature. To create great art, it seems, one must comprehend and consubstantiate human nature and aspiration. And if

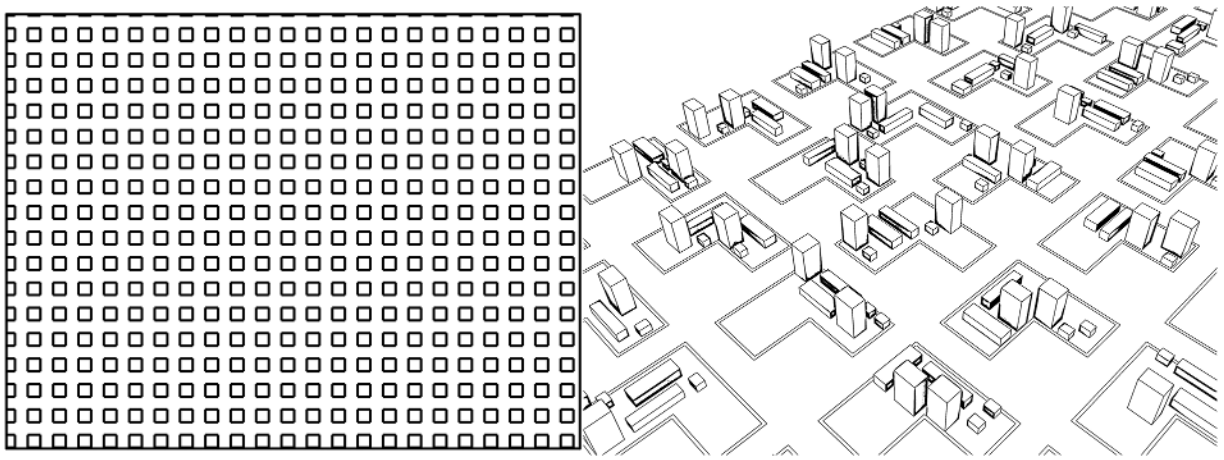

Figure 1.1: Design automation versus design intelligence.

no computer can create great art, then computers cannot be true artists. In other words, a computer cannot create architecture because it has no legitimate human essence or experience to draw from.

Against this it can be argued that computers can, and do, create great works of art. It is only a matter of time before critical tastes understand the esthetic value of generative computer art and elevate it to eminence. It is not completely facetious, and it may in fact be polemical, to point to non-fungible token digital artifacts as being valuable pieces of art under some interpretation of the adjective "valuable".

Another argument sometimes heard against intelligent computational design is that "design" is what architects enjoy doing. The use of computers should be restricted to tasks that architects do not enjoy such as fitting out washrooms. This is of course subjective; there are without doubt architects who enjoy designing washrooms or doing whatever task it is thought the majority dislike. In any case the claim is normative rather than a positive statement of fact. Researchers can simply remain agnostic as to what real architects would like to see happen. The research question is not "should we build computers that can create architecture" rather it is "can we build computers that can create architecture".

Ultimately, intelligence in design should not be seen as a supplanting but rather, as a supplementing technology. Architects should learn to take advantage of the advances in analytical decision making that parametric design methods provide. They should not fixate on the man versus machine conflict that may well arise in the future of design. Will architects go the way of the assembly floor worker whose role has been rendered all but extinct by the automation of machines? This, it must be said, is possible. Indeed, our present-day observation of the portents of technology suggest that this is possible for almost any human profession. However, it is unlikely to happen any time soon; architects do too many things other than create building geometry to be so easily replaced.

The evolution from automation to intelligence in computational architecture occurred in roughly four stages: Computer Aided Design (CAD), Building Information Modeling (BIM), Computational Design and Parametric Design. These stages were not linear and sequential; they overlapped, co-existed, and were in some cases complementary. CAD was

mainly a 2D drafting paradigm; BIM, and Object CAD before it, extended 2D drafting to 3D modeling and introduced some object and relational intelligence; Computational Design made it possible to algorithmically solve general design problems as well as complex geometric problems, while Parametric Design took advantage of advanced algorithms, data analytics, and artificial intelligence to create intelligent design.

In this chapter we will review the evolution of CAD and BIM as architectural design paradigms. This will contextualize our discussion of computational design intelligence. In the next chapter we will review project cases of both the Computational Design and Parametric Design paradigms.

Computer Aided Design

In a discussion about intelligence in design, the acronym CAD seems like a misnomer. Traditional CAD products like AutoCAD and Rhinoceros, without plug-in enhancements, are not thought to be aiding design, per se, in any particular fashion. Rather they are drafting and modeling automation tools that help the designer visualize and represent design elements (Figure 1.2). The design agency remains exclusively in the hands of the designer. It would perhaps be better, as sometimes happens, if the acronym CAD referred to Computer Aided Drafting.

The origins of CAD are traced back to military applications during and after the Second World

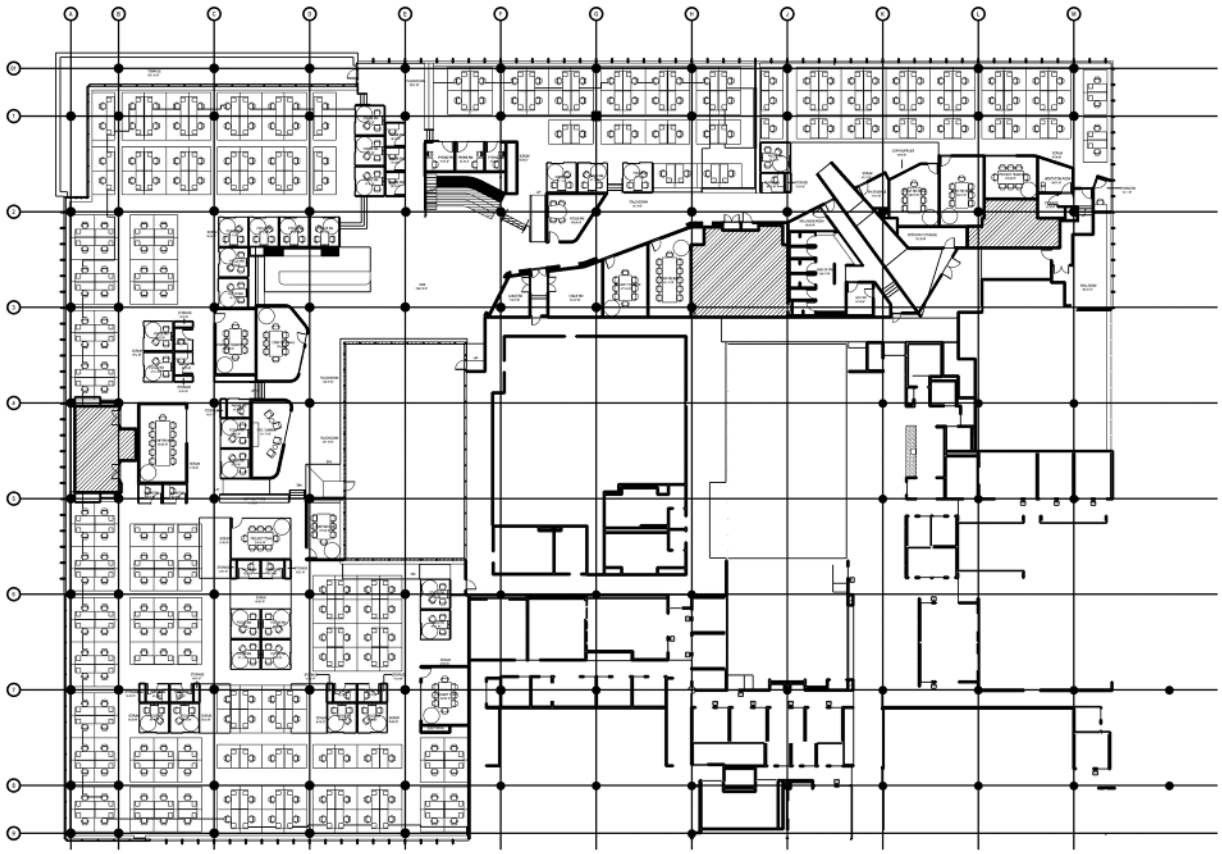

Figure 1.2: CAD drawing.

War. Applying machines to make the production of drawings more efficient began with the engineering disciplines. These early machines not only improved the process of drawing production they also facilitated numerical computations associated with drawings. Early computer developments in the 1940s were thus influenced by military applications where the machines were used to calculate things like ballistic trajectories. As depicted in the film *Hidden Figures (2016)* the term computer was, in fact, initially used to refer to the people who performed engineering calculations.

Thereafter, the development of CAD from the 1950s to the 1990s – after which newer technologies began to be ascendant – was a confluence of industrial enterprise, software enterprise and academic research. Once corporations began to take greater interest in computation from the 1950s, business automation companies like International Business Machines (IBM) began developing machines that could service these needs. By the 1960s the need for computer and graphic systems to improve engineering productivity was starting to emerge in large manufacturing companies.

This was seen particularly in the automotive industry (for example, Renault and Ford) and the aerospace industry (for example, Lockheed). These companies built in-house computer systems for their internal use and in so doing triggered a broader interest in such systems both in industry as well as in academia. For example, Ford developed the PDGS system, Renault developed CATIA, and Lockheed developed CADAM.

An important component of computational design systems is their hardware requirements. Graphics computers must have powerful computational units to perform intensive graphic calculations as well as powerful graphic units to render complex images quickly and legibly. The early CAD applications were run on large mainframe computers from vendors like IBM with purpose built graphic display systems.

The need of the manufacturing industry to represent complex curves and surfaces made it necessary to involve academics in the development of CAD systems. Mathematicians and computer scientists were instrumental in developing methods for representing complex curvilinear geometries. Notable contributions were made by Pierre Bezier of Renault who proposed Bezier curves and surfaces in the 1960s, and Rich Risenfield and Ken Versprille, both from Syracuse University, who in the 1970s developed B-Splines and Non-Uniform Rational B-Splines (NURBS) respectively.

Motivated in part by the entertainment industry, which began to use computer generated models in movies, the CAD industry began research into 3D modeling in the 1970s. The CAD Group at Cambridge were particularly active; one of them, Ian Braid, is credited with developing the Boundary Representation method (B-Rep or brep) while the group as a whole is credited with developing the ACIS solid standard in the 1980s. This 3D modeling would become the foundation for future paradigms like Object CAD and BIM. While many of the 3D modeling products were targeted at architecture and engineering disciplines, some like 3D Studio Max, Maya and Blender were targeted at the media and entertainment industries.

The 1980s saw important changes that both marked a high point in the CAD paradigm but also marked the entry of new software developers who would proceed to transform CAD into new approaches like Object CAD and BIM. At the hardware level the industry moved from 16-bit mainframe computers to 32-bit mini-computers to engineering workstations and ultimately to the epoch-making personal computer. At the software development level new entrants like Autodesk (authors of AutoCAD) and Parametric Technology

Corporation (authors of Pro/Engineer) would ultimately proceed to usher in new paradigms with products like Architectural Desktop and Revit.

AutoCAD (Figure 1.3) and Rhinoceros (Figure 1.4) are two of the most important CAD products ever developed with substantial market share in the architectural, engineering and manufacturing industries. They both encapsulate the key features of the CAD paradigm and have both been the foundation for newer paradigms and technologies. In the 1990s Autodesk moved to aggressively develop a suite of 3D Object CAD products built on top of the AutoCAD platform. These included products like Architectural Desktop, Mechanical Desktop and Land Desktop for architecture, mechanical engineering, and survey/civil engineering respectively. In the 2000s Rhinoceros, on the other hand, became the platform for the most popular visual scripting computational design interface – Grasshopper.

The CAD paradigm is characterized by key functionalities including drafting tools, drawing organization tools, and drawing management tools. Drafting tools comprise of geometry creating commands, geometry transforming commands and geometry editing commands. The key geometry creating command is the "Line" command. Other key geometries are points, rectangles, arcs, and circles. More advanced versions of CAD software include advanced geometry commands like polygons, ellipses, and splines among others. Geometry transformation tools include translation commands like "Move", "Copy" and "Offset"; reflection commands like "Mirror", as well as "Rotate" commands. Geometry editing commands include "Trim", "Fillet", "Chamfer", and "Extend".

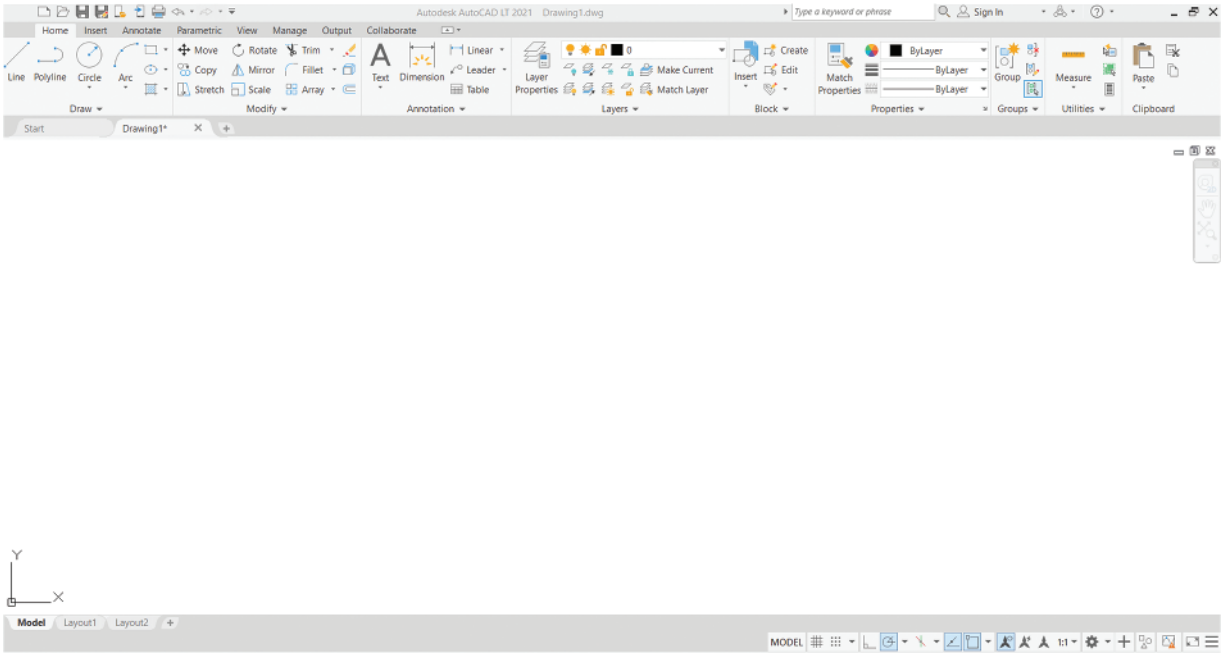

Figure 1.3: AutoCAD interface.

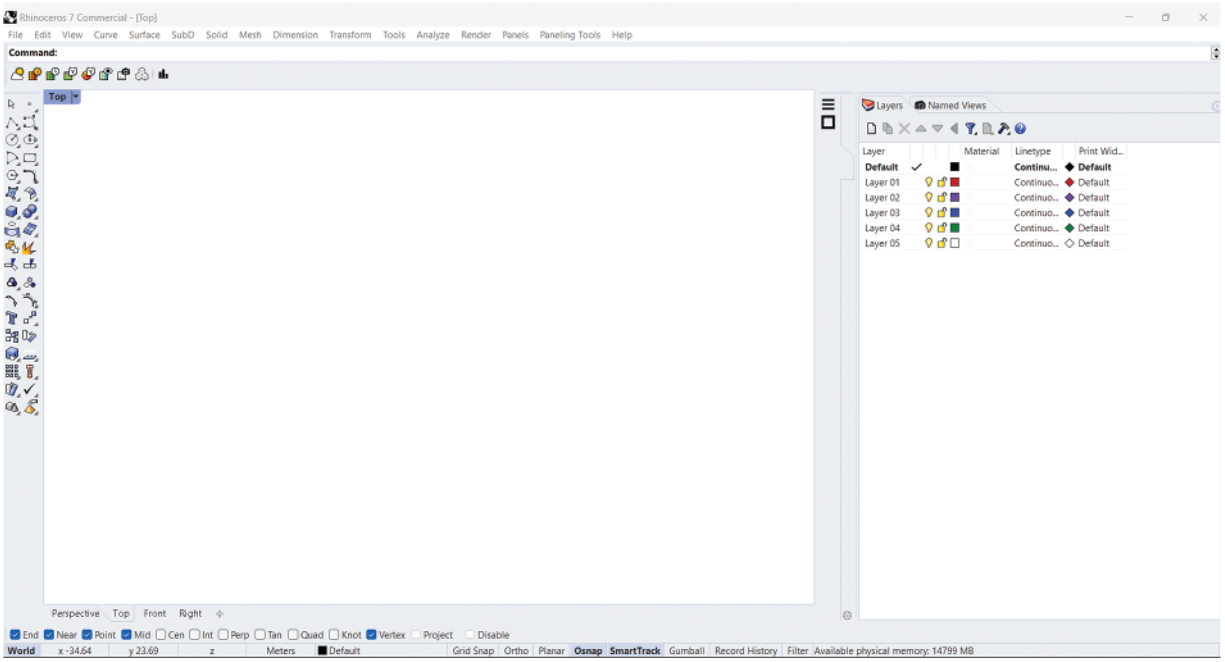

Figure 1.4: Rhinoceros interface.

The key drawing organization feature of CAD programs is the layer structure. Layers are auxiliary objects that act as containers for geometric objects. Any given object belongs to a single layer. This enables users to organize drawings according to hierarchies of overlaying geometry. This is invaluable for managing complex drawings with large amounts of overlapping information. Layers can typically be visible or invisible, locked or unlocked for editing, or active or inactive. Layers can also be used to control the visual properties of the objects they contain such as line type, line style and line color.

Drawing management tools include layout viewports, that help users organize sheet views for plotting; external references and blocks, that allow users to reuse already existing content or create reusable geometry to represent frequently used objects; and block attributes for assigning parameters to geometry. In addition, there are a plethora of drawing aids that make tasks easier to perform such as object snaps that guide the drawing process, object filters that guide object selection, and user defined coordinate systems that make it more convenient to locate objects in space.

It is worth mentioning that though computer drafting is usually associated with 2D drawings the drawing space of many CAD programs is actually 3D in extent. This means it is not only possible to draft in 3D, many of these programs are also facile 3D geometry modeling tools. Thus, many CAD tools include commands like “Extrude”, “Loft” and “Blend” that enable the creation of both primitive and complex 3D surfaces and solids.

This can be seen as a first step toward Object CAD. 3D drafting or modeling of any objects other than geometric primitives can be time-consuming in CAD programs. However, if the desired objects have

been modeled and stored as parametric blocks then the future creation of these objects is greatly simplified. CAD applications that are equipped with 3D blocks to facilitate 3D modeling are referred to as Object CAD. Object CAD programs were a precursor to BIM.

In retrospect, what can we say the benefits of CAD tools were when compared to the manual drafting that they replaced? One obvious benefit was the automation of drafting tasks such as the creation of hatch patterns that we mentioned. Another obvious benefit was the ability to perform quick edits and updates without having to redo entire sheets of drawing. In the old days if a drawing needed too many changes, then at some point it became more practical to create a new one rather than deal with the mess of manually erasing ink from tracing paper.

The ability to reuse drawing elements rather than to keep redrawing them was another benefit of CAD over manual drafting. CAD, to a great extent, also standardized the look and feel of production drawing sets. This was seen as both a good thing and a bad thing. While standardization has likely improved efficiency in the construction industry with fewer errors arising from misinterpretation of drawing intent, many architects lamented the loss of character and individuality in CAD produced drawings. To many old school architects, their drawing style was their signature, and the loss of said signature was a loss of personal touch in their service delivery.

There is a technophilic view of the world that thinks that all computational advancements are good. This should not be uncritically taken as true. It should be remembered that most of the world's most beloved buildings were built without the use of computers. The use of computational design methods does not necessarily improve the quality of architectural design. In fact, the introduction of CAD as the primary tool of architectural design practice also brought limitations.

From an operational perspective, software technologies created a vendor dependence in the practice of architecture that many designers resented. Architects did not like to think their design choices and project management strategies were dependent on the product marketing decisions of commercial software interests.

From the perspective of practice culture, the introduction of CAD not only brought additional hardware and software costs it also introduced learning requirements for architects, many of whom never surmounted the new technology barrier. This dislocated many experienced practitioners from the design process, relegating them to onlookers and commentators rather than active participants in design. This dislocation also impacted the ability of these experienced hands to pass on knowledge to their apprentices through hands on mentoring. Without question, the rise of CAD, BIM and other software paradigms created a generational disconnect in architectural practice that is still playing out to date.

Ultimately, the inability to easily model in 3D, the inability to maintain coordinated document sets through the design and production process, and the lack of object or design intelligence features meant that CAD could not enter the new century as the predominant computational paradigm in architectural practice.

Building Information Modeling

The Origins of the BIM Paradigm

Chuck Eastman (1975) described a computerized building description system that had many of the features of current BIM applications, which he further developed in his book *Building Product Models* published in 1999. Van Nederveen and Tolman (1992) described the integration of building data with production models, which they referred to as

Figure 1.5: BIM model.

building information models. They can, as such, be credited with introducing the term BIM into popular use (Figure 1.5).

Software developers also played an important role in originating the idea of BIM. Autodesk, Bentley and Graphisoft were particularly active. In 2002, Autodesk defined BIM in terms of coordinated digital databases. The next year, in 2003, Bentley explained BIM in terms of a hierarchy of user needs that enable users to create, share and retrieve design information in a consistent fashion. In the same year Graphisoft identified the key advantages of Virtual Building (their term for BIM) as accurate 3D representations of design enabling visualization and automated documentation.

These definitions point to common intent in developing BIM technology. In particular all BIM solutions sought to be 3D representations of design with explicit object metaphors. Objects represented real world entities, such as walls, doors, and floors, complete with the object properties of real-world walls, doors, and floors such as thickness, area, and materiality. Objects also had object intelligence, allowing them to interact correctly with other objects just as they would in the real world. Thus, doors know that they belong in walls and walls know that they have specific interactions when they intersect other walls.

However, there were also different development paths and priorities for each software developer. Bentley and Graphisoft had BIM products (*Microstation Triforma* and *ArchiCAD* respectively) built on object-oriented CAD technology while Autodesk had both object-oriented CAD technology (*Architectural Desktop*) as well as relational database technology (*Revit*) which later turned out to be their BIM product of choice. Autodesk, given their comparative advantage in market value, preferred a proprietary approach with BIM as a centralized database technology built around their Revit product lines; Graphisoft favored a more open-source approach embracing industry interoperability standards like Industry Foundation Classes (IFC) as data exchange mechanisms; Bentley insisted on a federated database approach to BIM to leverage their existing *Microstation* line of products.

Additionally, BIM was strongly geared toward architectural production – the delivery of 2D drawings in the format and idiom understood by the construction industry. This, in fact, distinguished BIM applications from general purpose 3D modeling applications like *Sketchup*, *Maya* and *3D Studio Max*. There was also a keen awareness of the relationship between data and geometry, leading authors like Randy Deutsch (2015) in his book *Data-Driven Design and Construction* to describe BIM as a data driven technology. Finally, BIM supported a plethora of functionalities like clash

detection, photo-realistic visualizations, virtual and augmented reality, and building performance analysis which added to the tremendous power and appeal of the paradigm.

The Rationale for the BIM Paradigm

One of the main reasons given for developing the BIM paradigm was the need to improve productivity in the construction industry. In the early days of BIM two sources were frequently cited in the discussions of productivity in construction. The first was a 2001 paper by Teicholz (2001) that found that compared to all major non-farm industries in the US, the construction industry had failed to significantly improve productivity in three decades. The second was a 2004 report by the US National Institute of Science and Technology (NIST) that estimated the costs of poor interoperability in the construction industry to be US$ 15.8 billion (GCR, 2004).

It should be noted that there were detractors to these views. Rojas and Aramvareekul (2003) noted a discrepancy between macroeconomic data on construction industry labor productivity and microeconomic studies that suggested that the labor productivity in construction may have increased in the 1980s and 1990s. They attributed this discrepancy to problems with the reliability and validity of construction labor productivity calculated from macroeconomic data.

For example, data for the manufacturing industry is sampled from a few thousand establishment sources (manufacturing company accounts) while data for the construction industry needs to be sampled from hundreds of thousands of disparate projects with comparatively poor accuracy and consistency in the data. Rojas & Aramvareekul concluded that the uncertainty in the process of computing construction industry labor productivity makes it impossible to determine if it has increased, decreased, or stayed the same.

Another part of the rationale for the BIM paradigm in the 1990s was that adjacent industries had already demonstrated the power of object-based 3D modeling. Manufacturing software such as SolidWorks showed how parametric digital models could coordinate design, analysis, and fabrication within a single environment. At the same time, media tools such as 3D Studio Max made complex three-dimensional visualization more accessible and persuasive. Together, these advances helped legitimize the idea that buildings, too, could be designed as intelligent digital models.

The steady improvement of computer hardware was also a factor. Faster processors, larger memory capacity, better graphics performance, and declining storage costs made it increasingly practical to handle complex three-dimensional building models on desktop machines. Earlier generations of hardware had constrained digital design ambitions, but by the 1990s these limits were easing. As computing power became more affordable and reliable, integrated model-based workflows appeared technically achievable for everyday architectural practice.

Lastly, BIM was driven by the growing possibility of networked collaboration. As offices became more digitally connected, architects, engineers, consultants, and contractors could begin sharing information across computer networks rather than relying solely on isolated drawings and paper exchanges. BIM promised a common digital model that multiple participants could access, revise, and coordinate. In that sense, it aligned with emerging expectations that design work would become integrated, distributed, and collaborative through networked computing.

The Benefits of BIM

As already mentioned, the BIM paradigm, when compared to CAD, introduces features that are unquestionable innovations. These features include 3D modeling, object intelligence, enhanced coordination, parametric components, drawing production and data integration.

BIM-based 3D modeling (Figure 1.6) is an improvement on CAD-based 3D modeling because it is more efficient, and the outcomes are better visual representations of building geometry. BIM tools have explicit 3D modeling commands like "Wall", "Door" and "Window" while in CAD-based modeling such objects need to be handcrafted by the designer. BIM modeling commands also generate standard 3D component geometries compliant with industry representation standards.

The benefits of working in 3D are obvious. The main one is simply that we build in 3D and therefore representing design in 3D is more appropriate for the task. 2D representations of 3D design need more interpretation by the builder and this, presumably, leads to more construction errors. 3D models also enable a range of related benefits. These include design visualizations like photo-realistic rendering and animations, virtual and augmented reality, as well as clash-detection and performance analysis.

Geometric representations of construction objects in BIM are not purely graphic. They also carry embedded object intelligence that makes it easier to create these objects as well as to coordinate them in relation to other design objects. For example, in Autodesk Revit, the stair object is a complex geometric object representing, in both 2D and 3D, risers, treads, stringers, landings and other stair components (Figure 1.7).

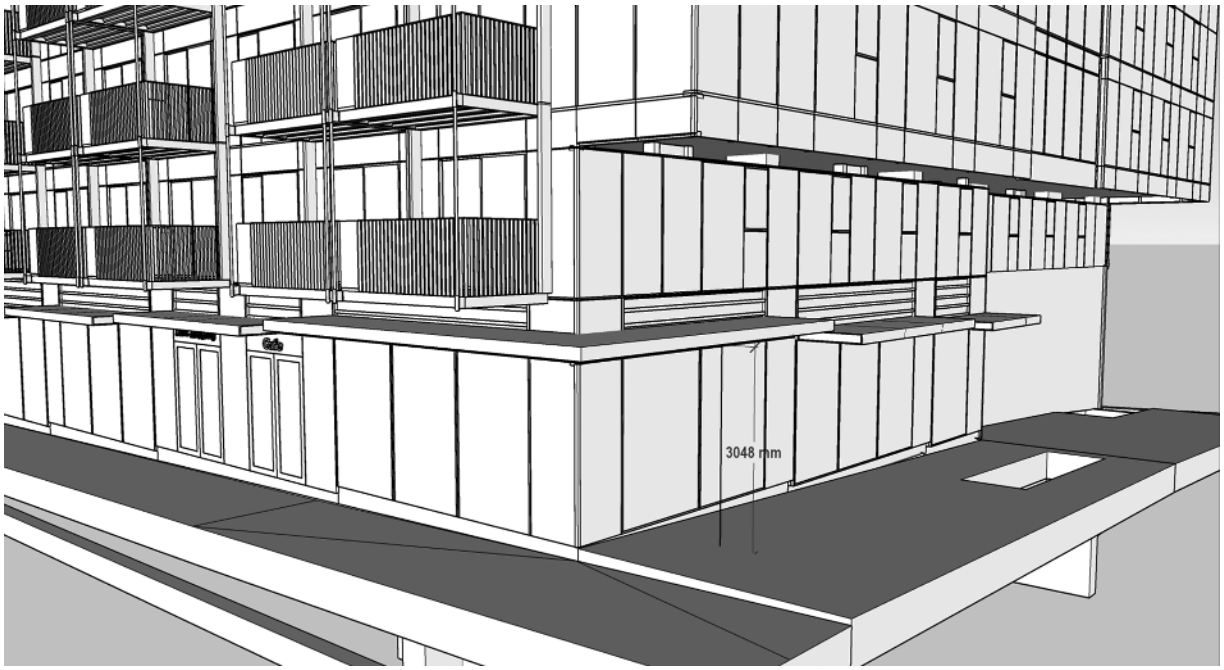

Figure 1.6: BIM 3D modeling.

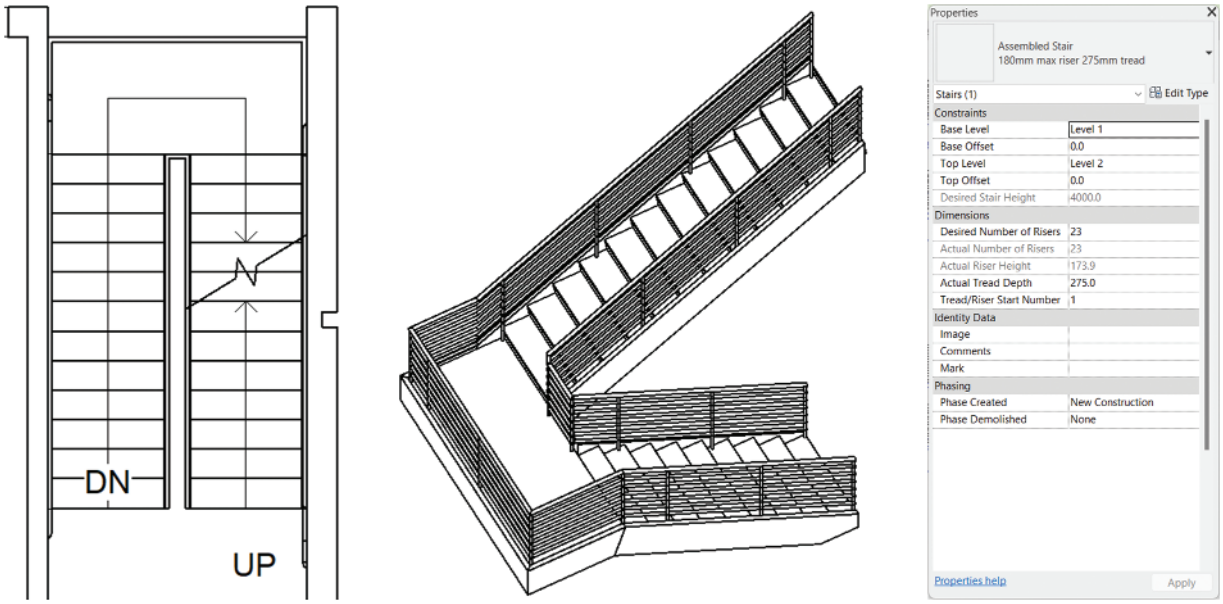

Figure 1.7: BIM object oriented parametric modeling.

In addition, the 2D representation can also include annotation elements like path arrows, tread/riser numbers, up and down text among others.

The stair object comes with parameters that can be geometric or non-geometric. Geometric parameters modify the geometric representation of the object such as its base level, its top level, or offsets from its base or top level. Non-geometric parameters carry important information about the object such as its assembly code and keynote reference.

The stair object further supports intelligent relationships with other construction objects such as floor slab and railing objects. Stairs understand that they can be hosted by a floor slab or that they can run between two floor slab objects. Railings understand their relationship to stair runs, stringers and landing objects.

Due to their object intelligence, the creation of stairs in Autodesk Revit is a much simpler process than their geometric representation suggests. It involves the sketching of run lines and the specification of about half a dozen parameters. This process, lasting a few minutes, generates a stair representation that is adequate for both design visualization and design documentation. In contrast, if the stair object were to be developed in a 3D capable CAD program it would take hours to model the stairs in 3D, draft a corresponding view in 2D, annotate the drawings, and then manually enter the stair specifications into a schedule table.

Architects do not work in isolation. The design process requires critical input from other experts, in particular consulting engineers. Of interest are structural and MEP (Mechanical, Electrical, Plumbing) engineers whose deliverables are design documents corresponding to the architect's design intent. These design documents must be coordinated into a complete, cohesive, and consistent set of instructions for the constructors to execute.

Coordination of consultants is one of the key roles an architect plays in the design process of a construction project. The design documents and specifications created by the consulting engineers must be brought together with the architectural documents and coordinated to ensure they are coherent. The ability to exchange 3D design models and reference these models into each other is of great benefit to the coordination effort. Moreover, the ability to perform clash detection exercises across different consultant models further enhances this capability (Figure 1.8).

As already mentioned, modeling in BIM is facilitated by taking advantage of object intelligence. This object intelligence is both geometric and parametric. Not only can designers leverage the object intelligence of standard components provided by the software (like walls, doors, windows, and stairs), but BIM tools can also empower designers to develop parametric object intelligence for themselves. This allows them to use object intelligence for custom design objects that are needed to fully express the design intent. This parametric component creation foreshadows computational design functionality that we discuss in the next chapter.

Creating the custom casework component shown in Figure 1.9 involves the following steps in Autodesk Revit:

- Set up a framework of construction lines (reference planes) to support component geometry.
- Dimension the construction lines and associate the dimensions with geometric parameters of component parts (like length, width, depth).
- Sketch sectional geometry to represent the different parts of the casework component such as the top, bottom, sides, drawer parts, etc.
- Lock the sectional geometry to the construction line framework. When the parameters associated with the construction lines are changed the parts geometry will also change.
- Use 3D modeling commands like "Extrude" to create 3D geometry from the sectional geometry.
- Dimension 3D geometry and associate the dimensions with additional parameters as needed.
- Create additional parameters such as material parameters for the different component parts.

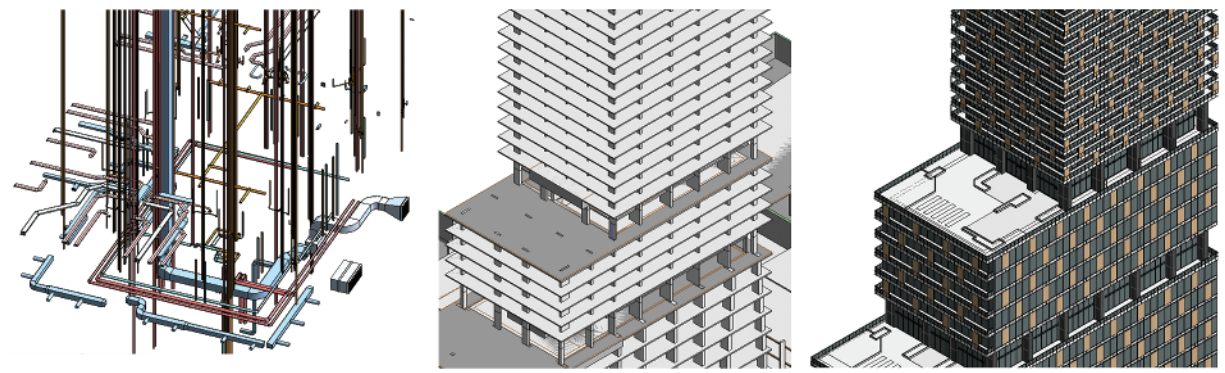

Figure 1.8: BIM consultant models for design coordination.

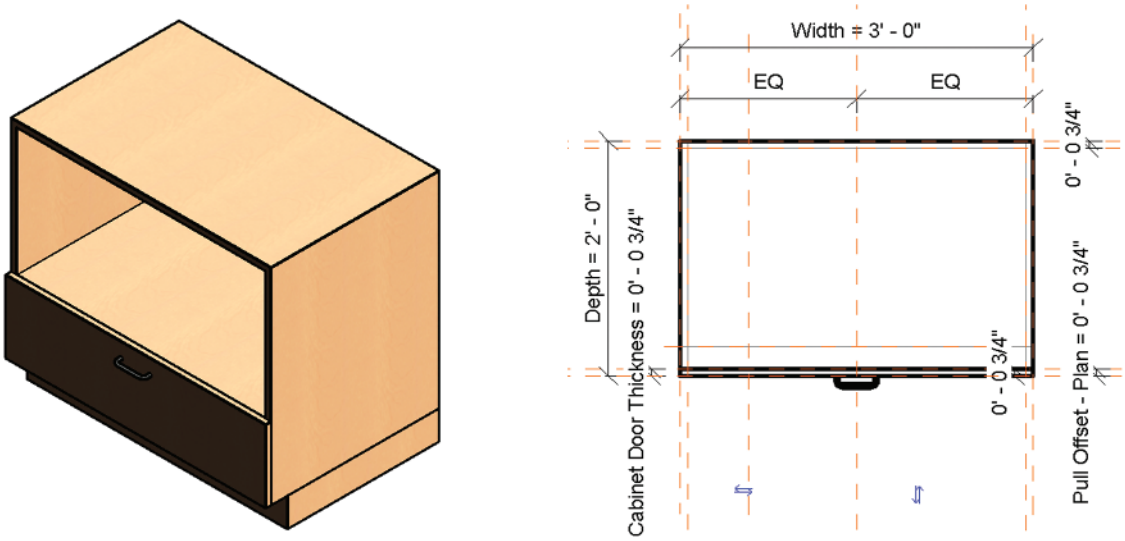

Figure 1.9: BIM parametric casework.

You can publish the parametric component into different projects that need it and within those projects designers will not need to recreate anything. They can simply change the dimensions and materiality of the casework component by changing the associated parameters. This ability to create parametric reusable components is another benefit of the BIM paradigm that was not present in CAD based workflows.

Despite being a 3D paradigm, BIM tools are still required to create 2D drawings that conform to the industry standards for construction documents (Figure 1.10). This is partly because the construction industry as whole, because of its decentralized nature, reacts to changes in digital technology more slowly than other industries. However, more importantly, not every component of a real building is modeled in 3D as this would be impractical. Some components and some information must still rely on 2D representation. In particular construction details are typically not modeled. They still rely on graphic language and 2D annotations to convey the complexity of materials, processes and geometry that characterize details.

Therefore, BIM tools typically have a full complement of drafting tools to enable them to augment 3D modeling with 2D content for the production of industry standard drawing sets. However, drafting in BIM tools is rarely done from scratch. Instead, 3D model geometry is leveraged as part of the 2D drawing creation process. This means that the 3D geometry created in BIM has special representation characteristics that allows for correct graphics when viewed in section or at high scales of resolution. This, as we have mentioned, distinguishes BIM modeling from modeling in general purpose modelers like Blender or 3D Studio Max. The latter do not have built in representations of modeled objects suitable for construction industry drawing standards.

Apart from geometric representation and object intelligence, BIM components also carry a wealth of embedded data. This data is accessed and manipulated using both the geometric and non-geometric

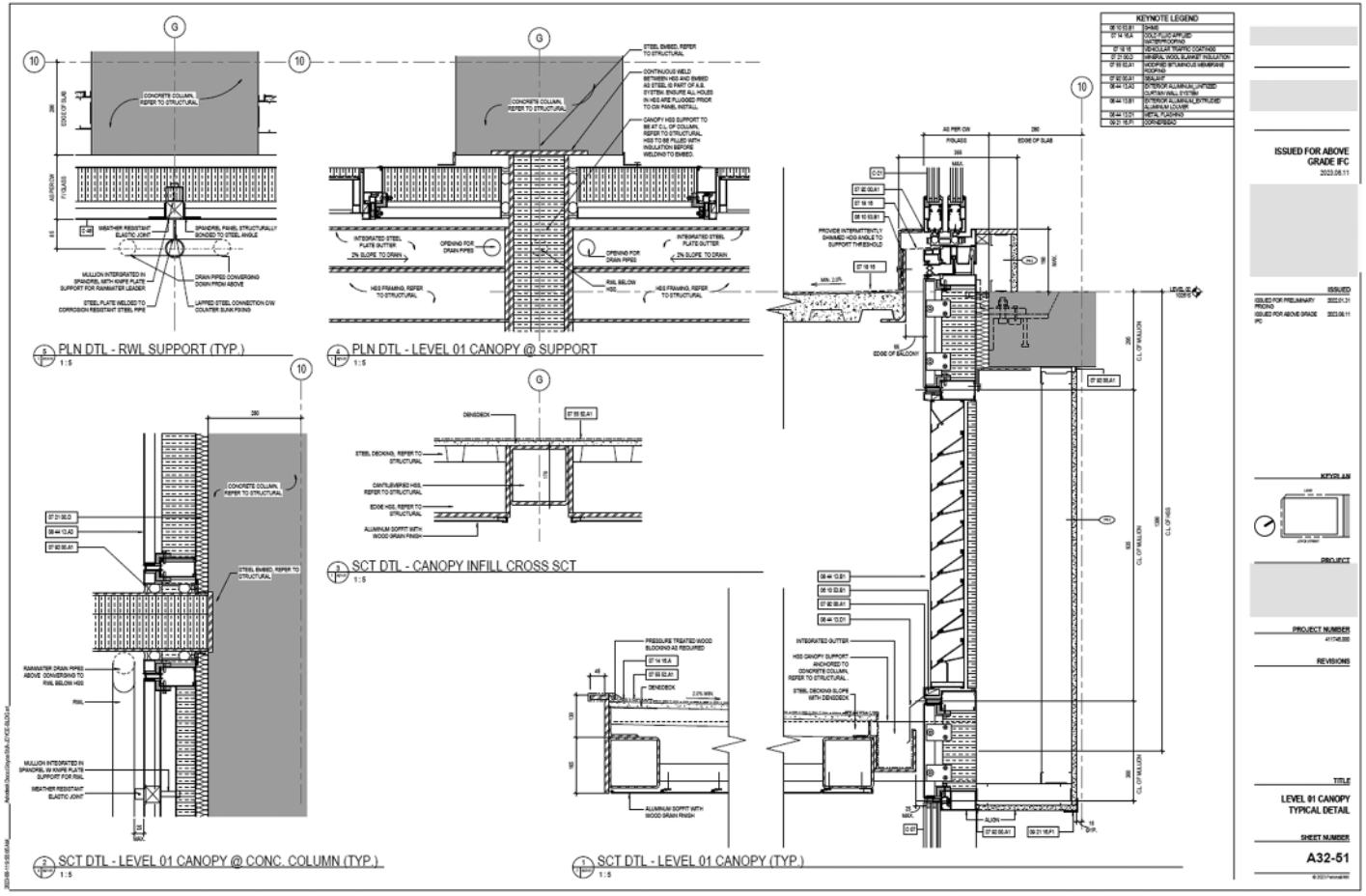

Figure 1.10: BIM 2D drawing.

Table 1.1: BIM schedule.

Door Schedule – Interior Suite											
				Opening Size		Panel			Frame		
Number	Name	Door Number	Fire Rating	Width	Height	Type	Material	Finish	Type	Material	Finish
4G01	Project Team Rm	4G01-1		3′-0″	8′-0″	FG			FT1		
4G02	Focus Rm	4G02-1		3′-0″	8′-0″	FG			FT1		
4G03	Phone Rm	4G03-1		3′-0″	8′-0″	FG			FT1		
4G04	Zone Rm	4G04-1		3′-0″	8′-0″	FG			FT1		
4G05	Zone Rm	4G05-1		3′-0″	8′-0″	FG			FT1		

parameters. For example, a door object has parameters associated with it's leaf dimensions, its frame dimensions, analytical properties like thermal resistance, supply chain properties like manufacturer and model number, among others. Designers can also create their own custom parameters for all BIM objects to address project-specific requirements. All this means that BIM models are deeply integrated with data and data structures.

What is all this data for? Some of it is needed to drive the characteristics of components such as changing their dimensions or material properties. Some of it, however, is needed to facilitate documentation. For example, BIM data can be used to automate the population of component schedules such as room schedules, door schedules and window schedules. On large projects with large numbers of components, automatic creation and coordination of schedules is a huge time saving and quality controlling benefit.

In conclusion, the BIM paradigm has endured and continues to be a transformative digital technology in construction. Both CAD and BIM have been important paradigms at the foundations of computational methods in architectural practice. As such, they provide useful context in understanding the newer paradigms like Computational Design and Parametric Design.