

**Building Information Twinning:
Investigating the Software Architecture of
a Building's Digital Twin**

Brad Williams

School of Architecture, College of Fine Arts,
Carnegie Mellon University

Master of Science in Computational Design
2023

CARNEGIE MELLON UNIVERSITY

School of Architecture
College of Fine Arts

Thesis

Submitted in Partial Fulfillment of the requirements for the degree of
Master of Science in Computational Design

TITLE:

Building Information Twinning:
Investigating the Software Architecture of a Building's Digital Twin

AUTHOR:

Brad Williams

ACCEPTED BY ADVISORY COMMITTEE:

Daniel Cardoso Llach

Principal Advisor

May 10, 2023
DATE

Jinmo Rhee

Advisor

05.10.2023
DATE

Abstract

This thesis critically examines the adaptation of Building Information Modeling (BIM) to receive real-time sensor data, bringing it closer to a representational paradigm known as 'Digital Twin' (DT) in the Architecture, Engineering, and Construction (AEC) industry. However, current monolithic BIM systems struggle with this integration, as demonstrated by the development of a construction LiDAR to BIM translation service. To address these challenges, the research proposes a microservices-based architecture with Docker containers, enabling efficient communication, adaptable data handling, and enhanced scalability. This thesis contrasts this novel approach against traditional methods, revealing its improved handling in dependency management and scalability. It concludes by advocating for a distributed, task-oriented BIM system that aligns with the increasingly collaborative AEC industry and modern software development practices.

I. Acknowledgments

My work is only made possible through the people who surround and support me, for which I'm fully grateful for.

Firstly, thank you to my principal advisor, Daniel Cardoso Llach, who welcomed me not only to the Computational Design program here at CMU, but to everything in this field I've developed a deep interest in. The wisdom and unwavering guidance towards fostering my interests have been the bedrock for this research.

I would also like to sincerely thank my secondary advisor Jinmo Rhee for the 2 years of dedicated support and vested interest in my work. His commitment to my research, his keen insights, and his ability to challenge my ideas have been instrumental in propelling this work to new heights.

Thank you to my cohort in the MS in Computational Design program for hours upon hours of provocative discussion, collaboration, and endless inspiration. The work and ideas are wonderfully insightful, and the dedication to expanding our field of inquiry is nothing short of phenomenal.

I'm also privileged to be within the Computational Design Lab (CodeLab) at CMU, and all of the opportunities for research it has provided. Thank you particularly to my teammates within the Rethinking Automation in Construction (ReAC) project, whose innovative work provided the initial impetus behind my curiosity in Digital Twin technology.

Finally and most importantly, thank you to my family for the continued support in my pursuits. My ongoing appetite for personal growth and learning is often unpredictable, and maybe even questionable at times. However the unrelenting support and encouragement continues to make everything I do possible.

II. Personal Statement

The writing of this thesis comes at a particular time in my career as an architect. Prior to my graduate studies at Carnegie Mellon, I practiced for about nine years, primarily working as a design coordinator on large commercial projects in the form of high rises and campuses. At the beginning of this time, as an undergraduate, there was a significant buzz surrounding Building Information Modeling (BIM) as the next great innovation for design coordination. Subsequently I, along with my colleagues, witnessed first-hand the adoption of BIM¹ as the industry standard, but more importantly, experienced the pace at which modern architectural practice can evolve around such innovations. Especially at the scale of large commercial development, young designers now are no longer draftspeople so much as they are data coordinators for teams consisting of hundreds of collaborators and consultants. As new technologies continue to integrate into practice, this data is not only increasing in complexity, but is visualized across an expanding set of mediums, including virtual reality, environmental simulation, and interactive web experiences.

Furthermore, the culture of how all professionals work in our modern era has undergone a swift realignment in recent years. Seemingly overnight during the 2020 pandemic, most architects made a shift from traditionally working alongside one another to collaborating entirely virtually. This is significant because it marked an instant demand for cloud-based data hosting and even tighter protocols surrounding information exchange and management. Arguably, the demand for digital management tools accessible to all project stakeholders in large projects and teams has never been higher, and the ability for professionals to adapt to these developments is paramount.

The practice of architecture and technology expectations from clients and broader society is undoubtedly morphing faster than ever, and it fuels a continued curiosity among my

¹Building Information Modeling (BIM) is a digital process and technology used to create and manage a digital representation of a building through a common organization of a project's data and parameters. It involves creating multi-dimensional digital representations of a building's physical and functional characteristics. This technology is further defined and elaborated on in Chapter 2.

generation towards where it may be going. This thesis is approached with a ‘near-future’ lens towards the practice of architecture. As a practitioner continuing to work in highly collaborative environments, I seek to discuss how tomorrow’s technologies may impact the practice of architecture today and introduce novelty not through expanding the technological capability, but through expanding the understanding of its application and influence on how professionals in our industry may interface with them. The better architects can understand the rapidly changing landscape of technology influencing their profession, the more effective they can become in servicing society at large.

Contents

1	Introduction	11
1.1	Motivation	11
1.2	Historical Context	12
1.3	Problem Space	16
1.4	Significance	18
2	Background	21
2.1	Data Formats within BIM and DT	21
2.2	BIM and Sensor Network Integration	27
2.3	Information Exchange In BIM and DT	31
2.4	Findings from Precedents Research	35
3	Hypothesis	37
3.1	Speculation	37
3.2	Methods Overview	40
4	Methods	43
4.1	Summary of Prototyping Technique	43
4.2	Construction Observation and Data Collection	44
4.3	Stud Detection Experiments	46
4.4	MVC Add-in Implementation	51
4.5	Microservice Implementation	55
5	Observations	61
5.1	Findings	61

5.2	Limitations	63
6	Conclusion	65
6.1	Discussion	65
6.2	Contributions	69
6.3	Next Steps	70

Chapter 1

Introduction

1.1 Motivation

Blueprints are no longer the only means of project communication in building design, construction and operations. Over the past half-century, the dominant method of design collaboration has transitioned from manual paper-based documentation to automated digital workflows. Virtual communication and control offers numerous potential advantages for design and construction management, including heightened efficiency in design production, diminished coordination costs, and enhanced project value through error reduction. However, since the 1970's, productivity in the construction industry has experienced a sustained decline while other assembly-based industries in the US economy such as manufacturing, have tripled or more [16]. Consequently, there remains a persistent need to innovate design communication technologies within architecture, engineering, and construction (AEC) practices [4, 38]. One of the primary reasons that productivity in the AEC industry poses a challenge stems from the need to coordinate a diverse group of experts for projects that are inherently distinctive. This demands effective collaboration and continuous problem-solving to ensure successful outcomes.

As an architect and computational designer, straddling the spheres of building design and technology, this demand to be innovative sparks curiosity. Specifically, I am interested in ascertaining the present trend of design collaboration, and the extent to which it can promote more efficient teamwork. I hold the conviction that critical examination of the

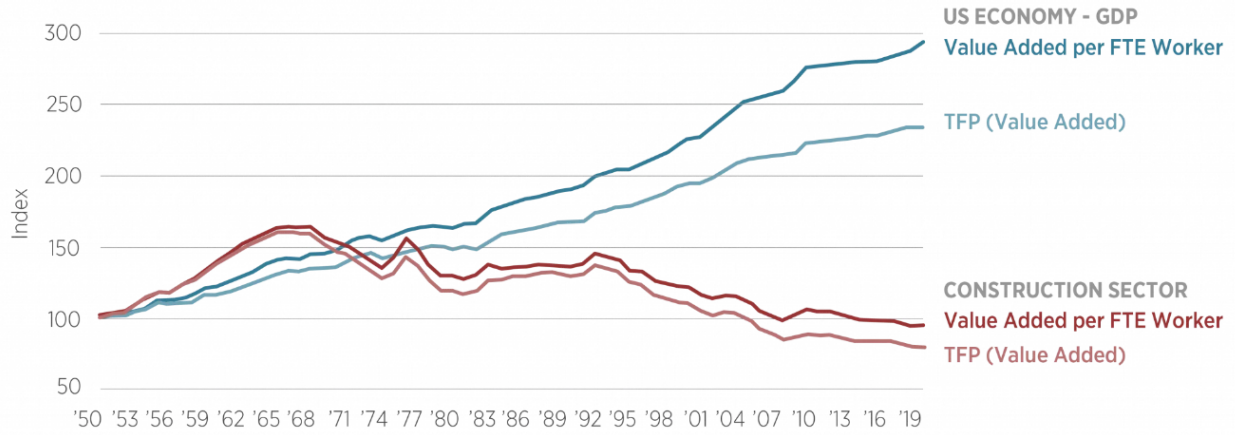


Figure 1-1: Construction Productivity Trend in the US (Goolsbee and Syverson 2023) [16]

instruments that facilitate our collaboration will equip us to work with better efficacy.

To commence my research process, in the following section (1.2 Historical Context) I briefly contextualize the trends of digital communication technologies leading up to today. Subsequently, in (1.3 Problem Space) I deliberate upon current trends and pre-existing challenges to establish a refined research question to investigate in this area of inquiry.

1.2 Historical Context

The objective of this section is to offer a presentation of the principal technological transformations in collaboration technology culminating in the contemporary era, thereby contributing to the formation and positioning of my research question. Certain technologies mentioned here are subjected to more in-depth analysis in (Chapter 2 - Background), subsequent to the establishment of their pertinence to my research endeavor.

The initial move away from paper blueprints in the 1960s ushered in the era of digital design with a technique called Computer Aided Design (CAD). CAD sought to translate the traditional paper drafting process to a digital platform, representing buildings as 2D drawings with graphic elements such as lines, hatching, and text annotations. This marked the first significant step toward automating the design process, though it was primarily a

reproduction of manual methods in a digital format.



meant moving beyond mere representations of design elements to creating models that could understand and respond to their relationships and contexts.

This convergence of architectural and computational thinking eventually led to the development of one of the most significant and widely adopted technologies for design and management within the AEC industry: Building Information Modeling² (BIM) [3]. Unlike CAD, BIM emphasizes 3D building objects as the basis for representing information. Since the 1980's (and popularized in the 2000's), industry practitioners have adopted BIM into standardized workflows, and in tandem, a large and diverse domain of research continues to grow surrounding its utility and future potentials as a dominant information technology paradigm for building planning, construction, and operations. [52].

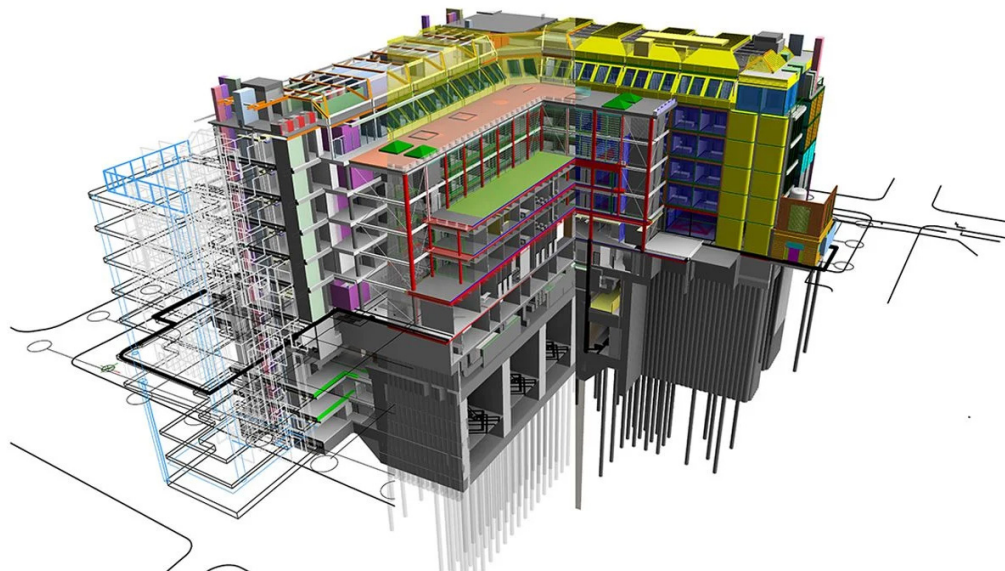


Figure 1-3: BIM Model (Scott Talon Walker Architects 2018) [42]

Despite BIM's proliferation, current research may suggest that virtual building representation will be expanded to a broad interconnected system of virtual services shared over

²Building Information Modeling (BIM) is a digital process and technology used to create and manage a digital representation of a building through a common organization of a project's data and parameters. It involves creating multi-dimensional digital representations of a building's physical and functional characteristics. This technology is further defined and elaborated on in Chapter 2.

real-time networks. These services may range from generative systems to automatically create geometry, to physical sensors to enhance the virtual design environment with real-world information. This real-time virtual representation is often called a building's Digital Twin³ (DT) and some researchers suggest a coming shift away from a BIM world view towards a DT world view.

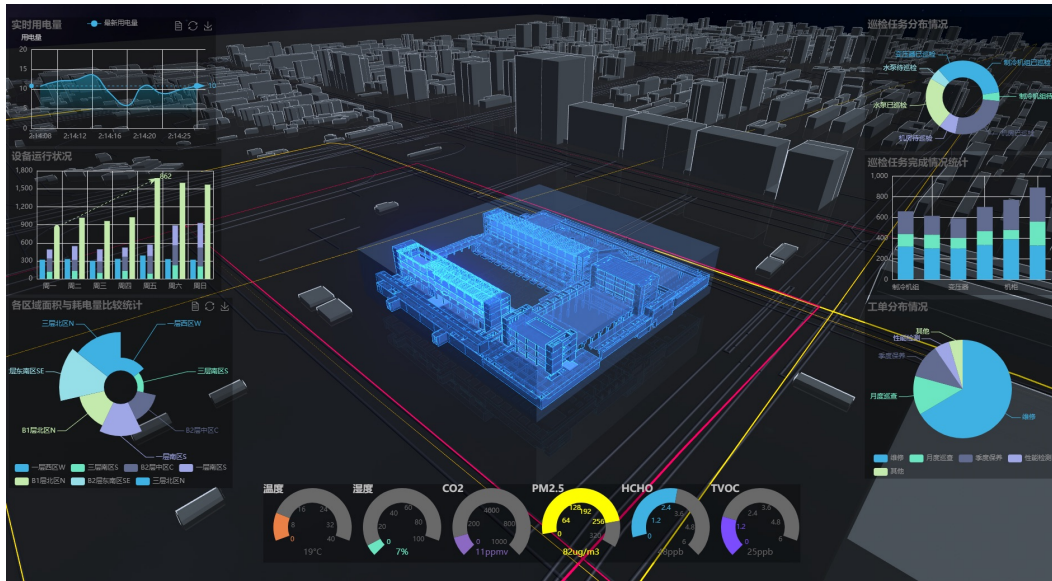


Figure 1-4: Digital Twin Interface (Villupuram 2020) [46]

With the growing influence of Internet of Things⁴ (IoT), Artificial Intelligence⁵ (AI), and data analytics technology on the AEC space, researchers are exploring potential benefits of these technologies such as enhanced predictability and analytics [5]. While BIM serves as

³Digital Twin is a broad topic that extends beyond building design, encompassing a larger technology paradigm that involves the creation of virtual representations of physical systems or processes. This topic is explored more in depth in Chapter 2.

⁴The Internet of Things (IoT) broadly describes a network of connected objects embedded with sensors, software, including devices and buildings. These devices can collect and exchange data with one another, enabling communication and interaction between themselves and users. IoT has received interest for its ability to enhance the environment with data, allowing for real-time insights and enhanced decision making.

⁵Artificial intelligence (AI) is a broad branch of computer science focused on mimicking tasks and processes typically performed by humans. AI systems for large sets of data include machine learning, deep learning, and reinforcement learning. Similar to IoT, automating tasks with AI may enhance data analysis for decision-making processes.

a digital simulation for documenting and managing the planning process, DT provides a one-to-one, real-time virtual representation of the physical building – which can be either fully or partially completed [32, 34]. Despite researchers’ different views regarding the exact distinctions between BIM and DT, there is clear consensus that DT represents an important paradigm shift in the AEC industry and is speculated to be the next critical technology and industry buzzword [34].

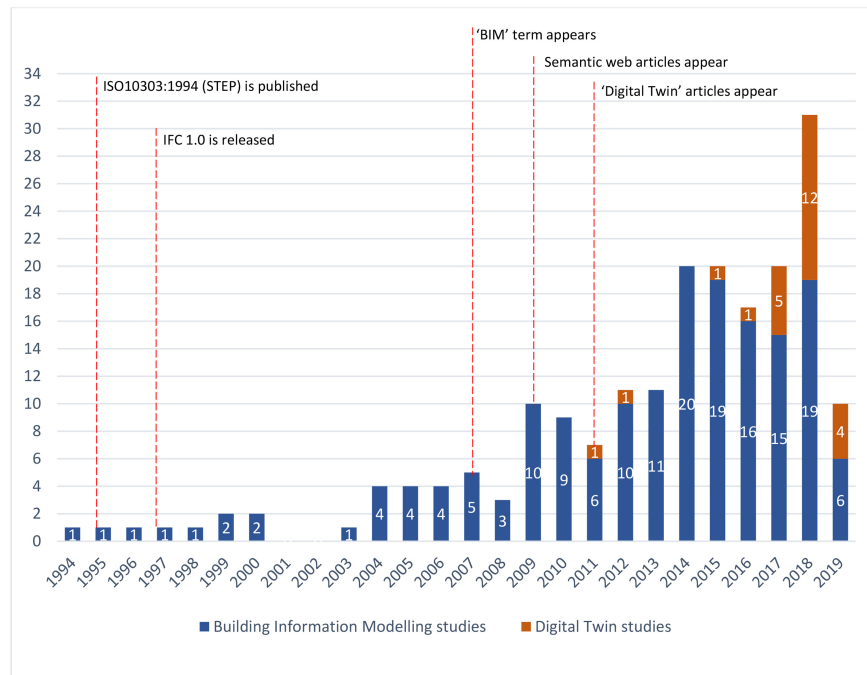


Figure 1-5: Distribution of Reviewed Papers by Publication Date With Important Milestones in the Construction Sector (Boje et al. 2020) [5]

1.3 Problem Space

In the digital progress of the AEC industry, the transition from BIM to DT has brought new opportunities and challenges. Although BIM has attained a dominant position as a technology for design and management, it is still beset by limitations in terms of interoperability and integration with emerging technologies such as IoT, AI, and data analytics.

Specifically, semantic modeling has been the focus of BIM research for decades. However, the level of collaboration between various systems and actors remains feeble [5]. Researchers observe and note that existing BIM capabilities lack the ability to effectively integrate with other systems or interpret different types of information, including sensor networks and other sources of information [29]. Conversely, DT promises a real-time, interconnected system of virtual services shared over networks, potentially improving the efficiency and effectiveness of the design and construction process. This is the core issue: DT necessitates a level of information collaboration that existing BIM capability may be incapable of supporting.

The technological novelty of DT in the AEC industry poses implementation challenges and is thus in its infancy [29]. At present, research on DT in the AEC industry primarily focuses on its potential applications, as demonstrated through prototypes and case studies [29, 26]. In these studies, where novel DT frameworks are proposed, researchers primarily implement DT capabilities with existing BIM platforms and methods [27, 30, 12]. However, there is a paucity of research on software system restructuring or data exchange methods between systems [45].

The issue at hand pertains to the transition from CAD to BIM, which involved a fundamental restructuring of the underlying software to represent buildings as objects rather than lines. As the AEC industry increasingly turns towards the adoption of DT as a potential new communication and representation paradigm, there arises a need for research that examines the underlying structure of the software architecture necessary to support this vision. Thus, the research question that this thesis aims to explore is: what constitutes the software architecture of a building's DT? If software architecture is defined as the organization of components and interactions of a system, I seek to understand what the components of a DT may be, and how they might interact.

Answering this question can yield insights into how the structuring of a DT's information may depart from an object-oriented BIM paving the way for new approaches. This research could uncover novel data structuring and exchange methods, enabling more seamless inte-

gration with sensor networks and resulting in enhanced decision-making in the design and construction process. Furthermore, this research may stimulate innovation in the development of new tools and software, helping the industry to transition from BIM to DT domains. Ultimately, I aim to contribute to the discourse on DT by drawing on the DT creation process, to better support its development as a viable technology for the AEC industry.

1.4 Significance

This thesis investigates the technical structuring of a building’s DT. However, the objective is not to investigate the implementation of a DT to merely advance it as the most recent technical paradigm. Rather, it seeks to gain a thorough understanding of a building’s DT software as a basis for relating the DT paradigm to broader trends in AEC collaboration. In other words, unveiling the mechanics of a building’s DT may demystify the technology beyond the trending DT buzzword. Since DT was developed in industrial settings outside of the AEC industry, it is important to remain critical towards its implications for architecture and construction practice.

For instance, BIM became a popular technology mostly because it supported the larger aspiration of greater integration between traditionally separated practitioners in the building design and construction process. During BIM’s adoption in the workplace, non-technical trends such as “big rooms” started to appear, where architects and contractors create large, temporary offices on project sites to work in the same space. In an almost diagrammatic approach, big rooms stitch together portable office modules, instead of delegating individual modules to each trade. Big rooms are highly effective for solving problems between trades in real-time, and literally eliminate barriers between typically separated offices. In the same regard, the notion of BIM as a single modeling environment where documentation activity occurs in the same digital space supports the parallel idea of “virtual big room.” It does not matter so much that BIM has extensive technical capabilities over CAD; the mere idea that BIM enables people to work in one virtual space is momentous enough to establish its

significance across the industry.

Understanding the software components of a DT can be significant as these findings might correlate to wider trends in the AEC industry, such as the integration of knowledge domains. Similar to the evolution from CAD to BIM, transitioning towards DTs doesn't only represent a shift in software capabilities, but it could also inspire or catalyze the way people work. Unlike manufacturing contexts which rely primarily on machines, the architectural design and construction process is a context which relies heavily on human collaboration and interaction. Just as the BIM paradigm fostered the idea of a "virtual big room," the DT paradigm offers yet another milieu for considering how people collaborate.

The vision of design and construction this research may help to advance is the shift towards tighter integration in a traditionally siloed industry. This shift is not a technological force, but a restructuring of project delivery methods⁶ in recent decades. In short, the industry is moving away from the traditional "design then bid then build"⁷ model towards more contemporary, collaborative models such as Integrated Project Delivery (IPD) and Design-Build (DB). These newer models strive to break down the traditional silos between different disciplines and stakeholders involved in the building delivery process. Instead, they promote the creation of a more cohesive and collaborative team that works together towards a common goal from inception to completion.

However, while the reformation of teams and the restructuring of contracts may be relatively swift and feasible, developing technology to support this more collaborative vision is a more challenging endeavor. BIM, as an illustrative example, has indeed paved the way for collaboration between multiple trades. Nevertheless, the overhead and skills required to participate effectively in a BIM environment can be extensive and may act as a barrier to greater technological participation. Despite its potential, BIM still necessitates formalities

⁶A project delivery method is a contractual arrangement among project stakeholders that defines the roles and responsibilities associated with designing and building a project.

⁷Known as "Design bid build," this is a traditional project delivery method in construction where the owner contracts with the architect and contractor separately during different project phases. Neither architect nor contractor work directly with one another

such as industry standard BIM Execution Plan⁸, highlighting that communication protocols can remain bespoke and complex from project to project. BIM's further limitations such as a high learning curve demonstrates a gap between technology and industry visions, despite a conceptual alignment between both. Thus, I investigate the technical aspects of DT in this vein; observing potential improvements to support the overarching trend of greater knowledge integration.

To further understand the technical context I wish to situate DT in, I conducted background research to support this question in the subsequent Background chapter. Elaborating on the historical influence, I first traced the development of data formats in both BIM and DT to understand their origins and differences. Next, I conducted a literature review of researchers currently utilizing BIM and sensor network technology in the same context to understand current trends in implementing their speculative frameworks. Finally, I researched current trends of information exchange to understand the changing technology of how people collaborate with design tools.

⁸A BIM Execution Plan is a contractual document that outlines how a building project will use BIM and the protocols for managing its data throughout its lifecycle.

Chapter 2

Background

2.1 Data Formats within BIM and DT

BIM is not a singular tool nor software, but rather a set of processes that employ a variety of technologies to create and manage a digital representation of a building, by means of a consistent organization of a project's data and parameters [13]. BIM is predicated on a structured system of project description rules that streamline communication between stakeholders and their corresponding digital tools.

One of the earliest and most significant computer systems for describing buildings was Charles Eastman's Building Description System (BDS) developed at Carnegie Mellon University in 1974. Eastman paid attention to the potential that a building's information could be represented as an object and an instance, as diverse information can be represented as hierarchical objects within a computer system. Thereby, his work departed from the conventional method of representing a building with lines and points, instead using objects to represent 3D building elements such as windows and doors [26]. The system aims to support the design, construction, and operations of buildings, making it useful not only for building design, but also for building lifecycle management [12]. In order to support these multiple applications, a consistent, hierarchical data structure was employed to describe the common building elements. Eastman's vision was to enable a large number of components to be assembled like an actual building, while remaining linked for efficient computational management [6].

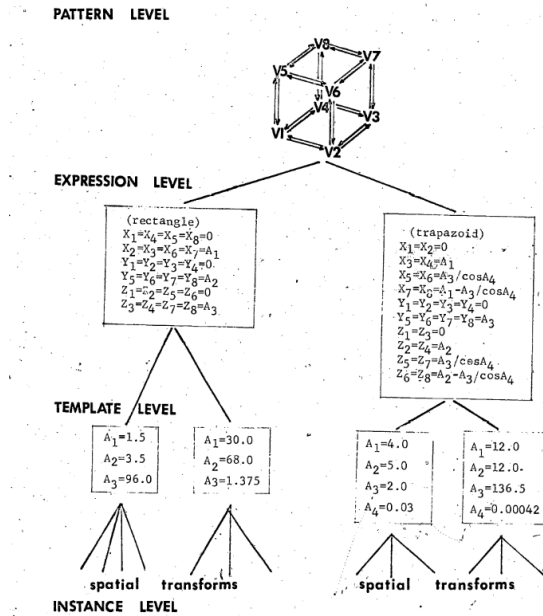


Figure 2-1: Eastman's BDS Depicting an Eight Foot 2x4, Door, a Rubber Motor Mount, and a Concrete Column (Eastman 1974) [12]

Furthermore, in the late 1970's, engineers from Liverpool University and later at Gollins Melvin Ward Architects introduced the Really Universal Computer Aided Production System (RUCAPS), which provided a shared framework for describing building components. RUCAPS built on Eastman's approach by linking building components through associative or relational means. Robert Aish, one of the engineers involved, later described the system in 1986 as "Building Modeling," which is regarded as the first use of the term in English now used in BIM [13].

In tandem with academic research efforts, the 1980's saw the emergence of several commercial products designed to provide Building Modeling for industry professionals. In accordance with Eastman's vision of a whole system, companies developed their own suites of modeling tools, complete with graphical user interfaces (GUIs), computational tools, and proprietary data formats. Among the most successful platforms of this era was Graphisoft's ArchiCAD, which incorporated the Virtual Building concept with a single binary data for-

mat, “.PLN,” for storing all project elements [27]. Other notable emerging platforms included CAdvance, which utilized a text-based “CV” file format, and Reflex, which used a binary-based “RDX” file format. Unfortunately, the competitive nature of the software packaging prohibited interoperability between platforms, hampering collaboration between systems. Moreover, data exchange was not considered necessary, as computers were perceived as mere workstations intended to complete specific tasks for individual workers [30].

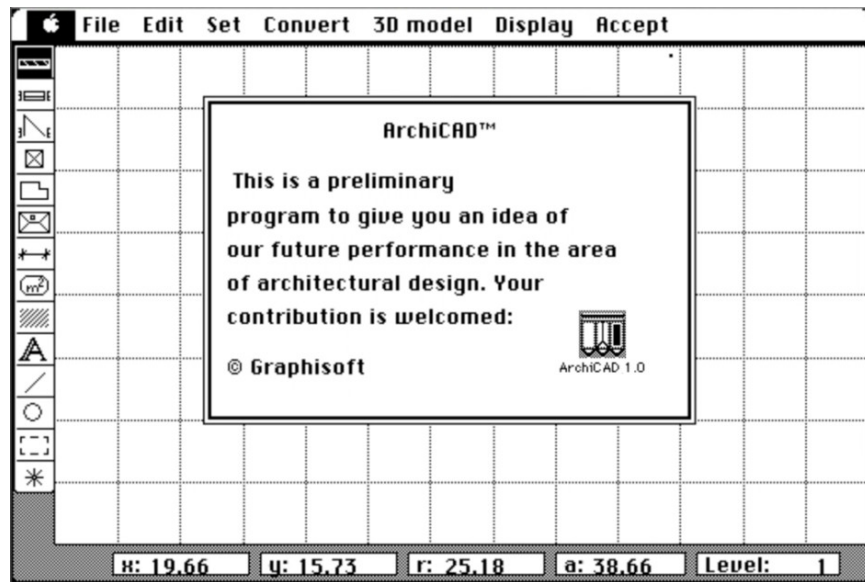


Figure 2-2: The Interface of ArchiCAD v1 on Apple Macintosh (Etiido 2020) [15]

The lack of a standardized computer modeling protocol for numerous manufacturing industries, including the AEC sector, prompted the International Organization for Standardization (ISO) to recognize the need for a universal open format [45]. In response, the Standard Exchange for the Exchange of Product Data (STEP) format was established in 1984. Complementing the STEP format, the EXPRESS language was developed as a text-based data model for the STEP format.

In the 1990s, a movement towards standardization emerged within the AEC industry as in response to STEP’s lack of speed and responsiveness to meet market demands [30]. To address this issue, the International Alliance for Interoperability (IAI), now known as

buildingSMART, was founded in 1995. The IAI spearheaded the development of Industry Foundation Classes (IFC), as an open, neutral data format for representing and exchanging BIM information that is widely used today.

In addition to the IFC format allowing for intra-software collaboration, the development of the Revit software platform in the 2000’s contributed significantly to BIM’s adoption at the turn of the century. Revit enabled multiple users to collaborate on the same building model, a technique inspired by version control systems in software engineering. This shifted the hosting of the “model” of BIM from a user’s desktop to a project’s central repository, which could exist on an offline local network on a server or be hosted on the web. By the 2010’s web-based BIMs became a popular workflow for developing projects in the AEC industry, inspiring the current trend of embracing DT technology in BIM. The coexistence of BIM models with real-time web networks made the exploration of integrating the two inevitable.

In contrast to BIM, DT is rooted in a distinct technology paradigm with fundamentally different origins. BIM and DT involve the creation of virtual representation of physical objects to simulate and facilitate real-world predictions and decision making [49, 36, 35]. However, BIM’s development was largely driven by research efforts in the AEC space during the emergence of personal computer systems in the 1970s. On the other hand, DT emerged from engineering and manufacturing contexts during the rise of networking systems in the early 2000’s. Consequently, the DT paradigm places an emphasis on simulating real-time representations of physical objects as they change and exist in a dynamic reality.

The term “Digital Twin,” was initially introduced in 2003 as part of a course on Product Lifecycle Management at the University of Michigan, and it has since been popularized by NASA’s use of the term to continuously simulate, monitor, and predict the behavior of critical equipment [49, 36]. The main components of a DT system include physical components, virtual models, and data that links them [5]. In the context of manufacturing, DT was developed in response to the need to manage large, interconnected systems of equipment. As

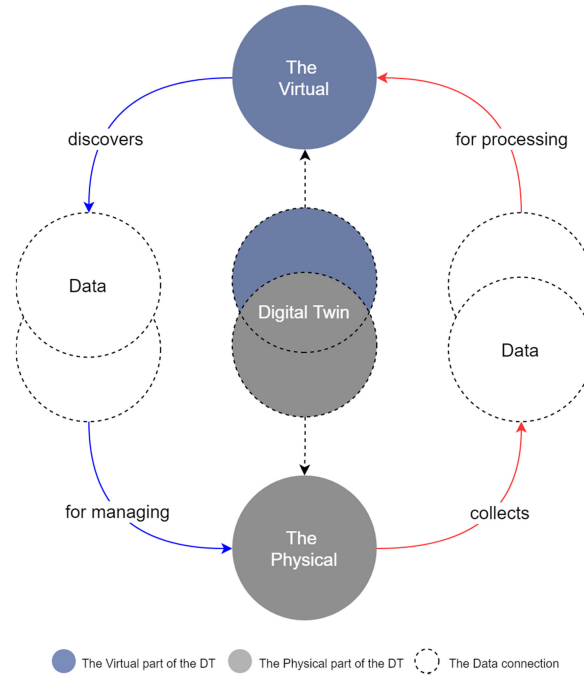


Figure 2-3: The Digital Twin Paradigm (Boje et al. 2020) [5]

equipment became more sophisticated with advanced sensors, data output, and communication with equipment, managing the aggregate network became increasingly complex. DT, sometimes referred to as a “system of systems,” replicates this network in a single virtual environment, enabling personnel to manage the complexity effectively [35]. While BIM was developed to aid in the planning of a static building, DT is primarily a tool for monitoring dynamic systems.

The desire to incorporate DT workflows within the AEC industry has been primarily motivated by the need to simulate building operations in order to inform efficient operation, management, and maintenance. Similar to machines in a factory, an operational building can be viewed as a collection of systems supporting environmental control mechanisms, such as heating, cooling, and lighting. Researchers have demonstrated that the benefits of DT workflows are particularly useful for supporting decision-making in building operations performance and maintenance. For example, Wang et al. have exemplified a building facility

management interface called the “digital twin-based framework for green building maintenance system (DT-GBMS)” with real-time photos captured by robots and performance data from sensors [48].

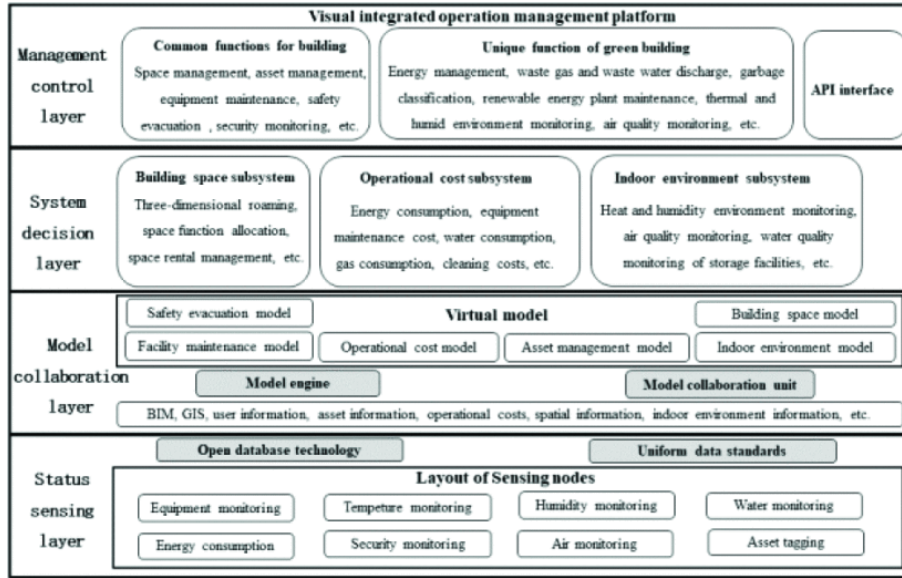


Figure 2-4: Digital Twin-Based Framework for Green Building Maintenance System (Wang et al. 2020) [48]

Researchers have also begun to identify the benefits of DT technology in the design and construction phases of a building’s life cycle. Specifically, some researcher’s work utilize a dynamic geometric design model that is in sync with the physical parts, and a real-time data workflow from the artifact to the model that updates the model according to physical changes [14]. For example, Kaewunruen and Xu have shown the utilization of 3D building information combined with cost and emissions data to run construction simulations of a railway station project.

Numerous conceptual frameworks have been proposed for integrating BIM and DT in operations, design, and construction to facilitate monitoring and simulation. However, these frameworks are limited to providing only a conceptual and diagrammatic basis for the application and feasibility of DT technology in the context of BIM software. To gain a deeper

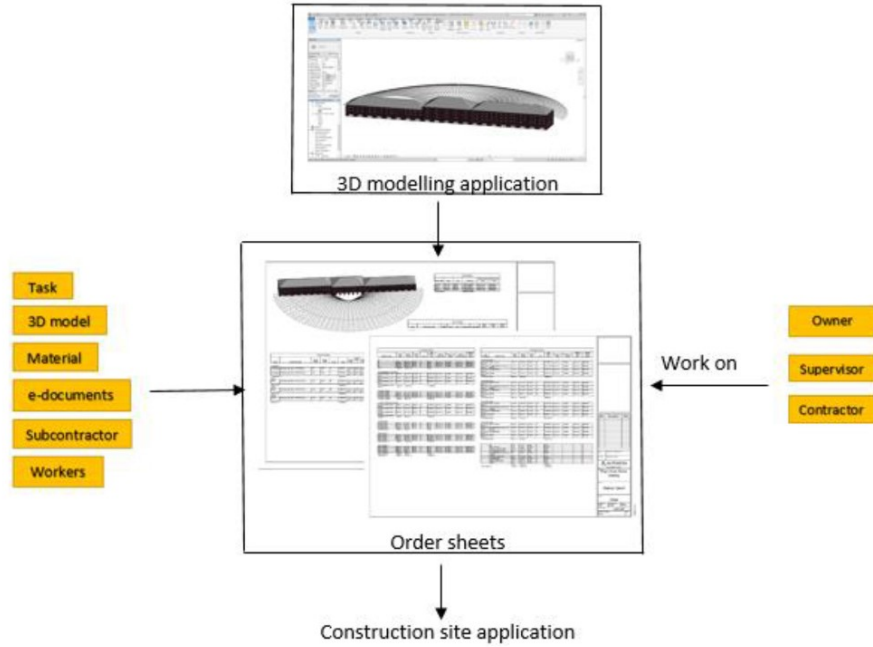


Figure 2-5: Real-Time Construction Simulation Integrating Real-Time Data With a 3D model (Kaewunruen and Lian 2019) [22]

understanding of my research question —what is the software architecture of a construction digital twin?, it is essential to explore how these frameworks are built and implemented beyond the scope of framework diagrams. In other words, many of the precedents and diagrams presented above merely show arrows as data flows between BIM and DT, but do not provide technical insight towards how sensor networks are integrated into geometric BIM models. In the following section I aim to expand my background research by investigating this integration further.

2.2 BIM and Sensor Network Integration

To delve into the potential software architecture of a building’s DT, a review was conducted on previous attempts which implemented DTs using BIM. The purpose of this review was to identify common trends and patterns in how researchers build a DT using a BIM model. The literature surrounding DT for AEC applications is extensive, and I approached my

background research of previous attempts via published systematic reviews. In 2021, during the onset of my research, researchers at the University of Michigan led by Deng et al. reviewed 123 papers on the evolution of intelligent building representations in the AEC-FM (AEC-Facility and Maintenance) industry and found that most of the implementations reviewed classify as real-time simulations supported by BIM [11]. Also in 2021, Researchers at Western Sydney University led by Opoku et al. identified 22 publications from their science mapping method¹ on Digital twin application in the construction industry and concluded that DT at its present state is synonymous to a BIM model within the construction industry [33]. In a subsequent 2022 literature review on DT adoption in the AEC industry by Opoku et al, they further noted that, while there are ambiguities between BIM and DT in the AEC industry, there is a formed subset of research which identifies BIM and DT as clearly separate technologies [34]. Similarly, in a literature review of DT’s applications in construction work-force safety by Hou et al, they conclude that Virtual Construction Simulation (VCS) uses BIM as an information base to extract the required construction information or geometric models, implying a separation of technologies [19]. Furthermore, Couprie et. al conducted a literature review for BIM-based DTs to improve maintenance procedures in buildings, and found that implementations were most successful using data extracted from the BIM of the building to create the DT [9].

Based on the literature reviews studied, the prevailing trend I found is that researchers may associate the BIM model at varying degrees when implementing a BIM-based DT; whether the DT is built from geometry extracted directly from the BIM model, the sensors are closely integrated within the BIM model, or the sensors are distinctly separated from the BIM model which receives updates.

Many researchers will use BIM as merely a source for geometry, without receiving further updates. According to Angjeliu et al., the geometry is the basis for building a DT and can

¹Science mapping visualizes a topic’s landscape by analyzing bibliographic data and mapping the relationships between their publications, authors, and themes, to identify trends and opportunities for further research.

be extracted from a BIM model [9]. Tchana also describes BIM as merely an accurate 3D model of the building whose components serve as the building blocks for their DT implementation [1]. Despite the different results and applications, these researchers emphasized the advantages of BIM in providing accurate representation of geometry. Even at an urban scale, Cureeton discusses how DTs are the fusion of extracted geographic information systems (GIS) and BIM data of buildings to construct digital replicas in virtual space [44]. The key trend here is the identification of a BIM model as the initial geometrical source for DT formation, and not a live component of the resulting DT which would change with real-time updates.

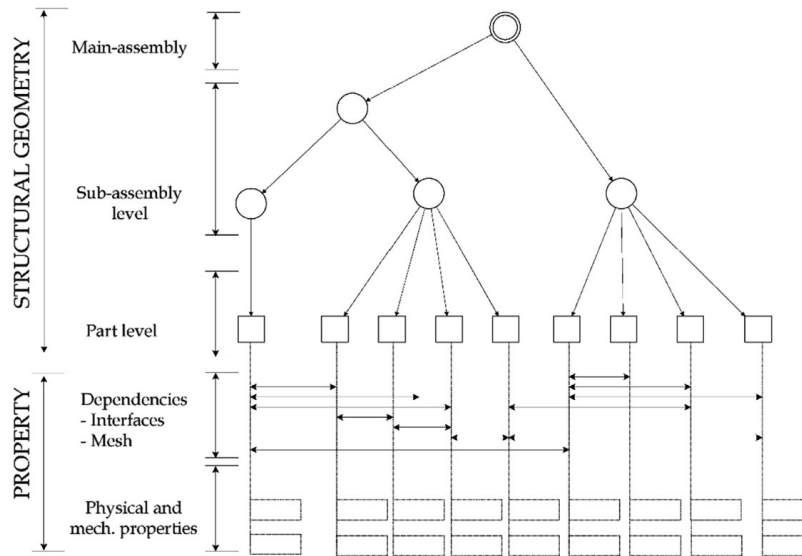


Figure 2-6: Schematic View of the Hierarchical Assembly of a Structural Model with Added Properties (Angjeliu et al. 2020) [1]

In contrast, the second prevailing trend from the precedents is that researchers will utilize BIM models by pairing them with dynamic systems which provide updates. This is fundamentally different from the first trend, because researchers engineer the BIM model to reflect the real-time updates. Jakobi et al. demonstrated a dynamic DT that continuously and automatically updated as conditions change [21]. Their DT addressed the operation

and maintenance (OM) phase to set up predictive maintenance services and minimize the downtime of heating systems. However, their DT paired a BIM model with an existing DT, keeping both entities separate but working together.

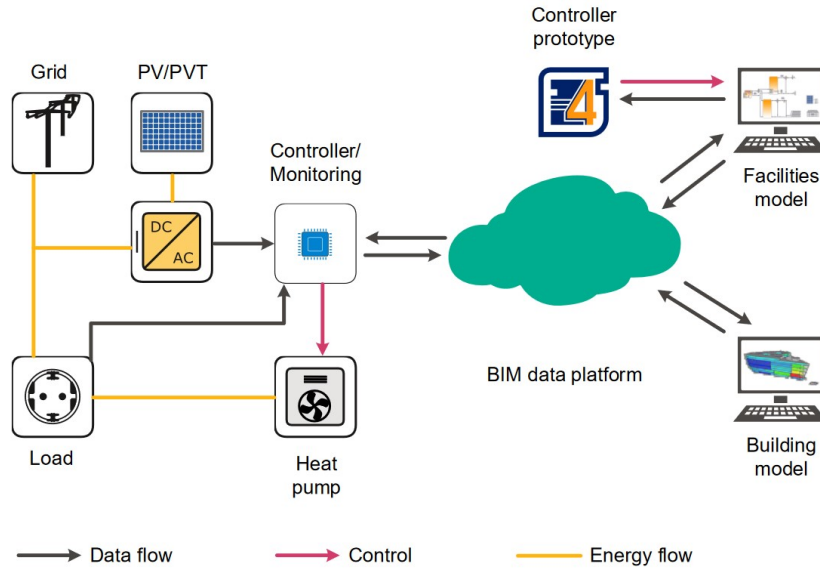


Figure 2-7: Separation of Sensor Network and BIM Platform (Jakobi et al. 2018) [21]

Similarly, Peng et al. reported a successful project case in a large hospital in China, which involved continuous and scheduled integration of static and dynamic data of more than 20 management systems from the design, construction, pre-OM phase up to the OM phase. A basic BIM was first built, and was used in conjunction with ontology information to create a DT [37].

Khajavi discusses the creation of a sensor network for the DT of a building to study the current technical shortcomings of establishing a DT. This fills the knowledge gap on where BIM stops, and another paradigm begins [23]. These studies show that researchers can build a DT by retaining the static BIM model and layering in dynamic data separately. By integrating BIM and sensor data, researchers can build a DT that is continuously updated and used to optimize building performance.

The literature review reveals that BIM and sensor networks can be combined in various

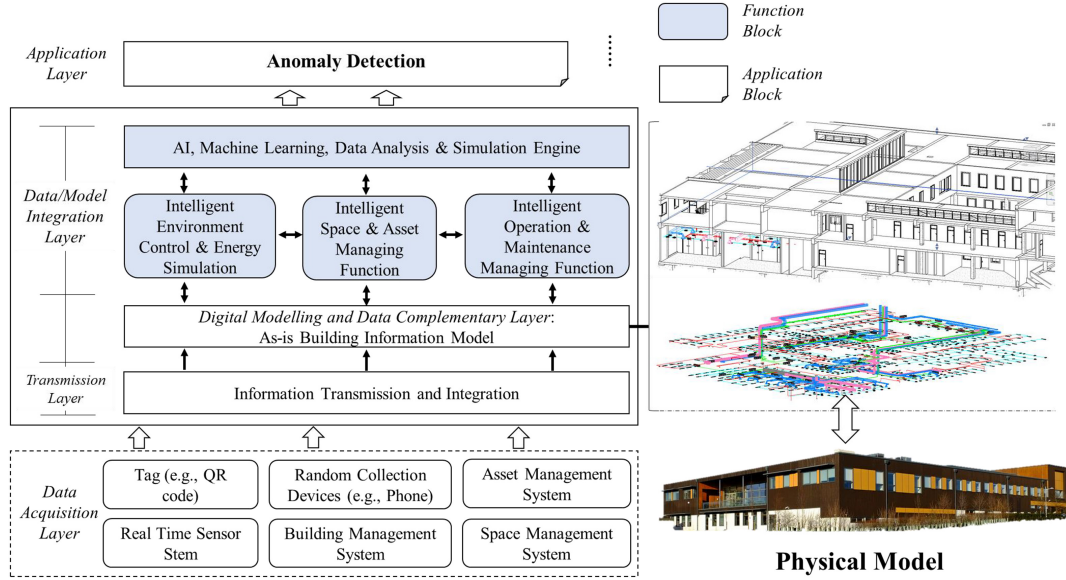


Figure 2-8: DT Framework Highlighting the Separation of “as-is” BIM and the Sensor Network (Lu et al. 2021) [29]

ways, with the precise structuring of these technologies remaining an active area of research. However, there is still a noticeable gap in understanding the technicalities of integrating sensor networks with geometric BIM models, which can lead to uncertainties when structuring a building’s DT beyond experimental prototypes. This may be attributed to the fact that much of the reviewed literature is developed from an "AEC-centric" perspective, focusing on the novelty of the solutions rather than addressing data concerns. As a result, further research into information patterning of a DT from a broader perspective is desired in order to bridge these gaps and provide a more comprehensive understanding of the integration between BIM and sensor networks.

2.3 Information Exchange In BIM and DT

Efficient information exchange is critically important for the integration of BIM and sensor networks. Therefore, I examined existing exchange formats for BIM and real-time exchange methods for web-based services. The goal was to understand the current limitations of in-

formation exchange between BIM and network systems. First I investigated how the IFC schema has adapted since its pre-web inception in 1995. The IFC schema has been considered a flexible exchange method because it can encode BIM data in multiple data formats depending on the application [42]. The main data formats for exchanging IFC data files between BIM applications are '.ifc', which utilizes the STEP data format, '.ifcXML', which uses the eXtensible Markup Language (XML) data format, and '.ifcZIP', which is a compressed file format that uses the ZIP file format to compress the IFC data files [8]. Of these formats, only XML is considered native for web communication. However, XML's inefficiencies in reading and parsing ² data during service execution has made it less popular in web services [44]. In contrast, the JavaScript Object Notation (JSON) format has gained popularity because of its faster parsing efficiency and lightweight nature [10, 21]. JSON is a key-value format, lightweight, independent of any specific programming language (despite JavaScript in the name), and is text based, making it human readable.³ Furthermore, JSON's optimized parsing and data generation capabilities combined with enhanced readability, facilitates cooperative environments for both humans and machines [47]. Recently, researchers have begun to conceptualize the IFC schema in the JSON data format beginning in 2017.

In response to these web-based trends, buildingSMART, a leading influential group in industry standardization, has published beta IFCJSON data formats and parsing tools while continuing to conduct research and development. They have identified a key objective of exploring scalable and widely adopted data exchange solutions that can work with a broader

²arsing is the process of analyzing a string of symbols according to the rules of a formal grammar to determine its grammatical structure. It is commonly used in computer science for interpreting and processing structured data.

³Compared to EXPRESS, which is the underlying data structure used in STEP file format, JSON is considered more human-readable due to its simplicity and intuitive structure. EXPRESS is a formal language with a complex syntax that requires specific knowledge and training to read and understand. On the other hand, JSON uses a key-value pair structure that is more easily understood by humans and can be interpreted without the need for specialized training or knowledge of a formal language. Additionally, JSON is less verbose than EXPRESS, which means that it is easier to read and comprehend, making it a more accessible option for users who are not familiar with the particularities of EXPRESS.

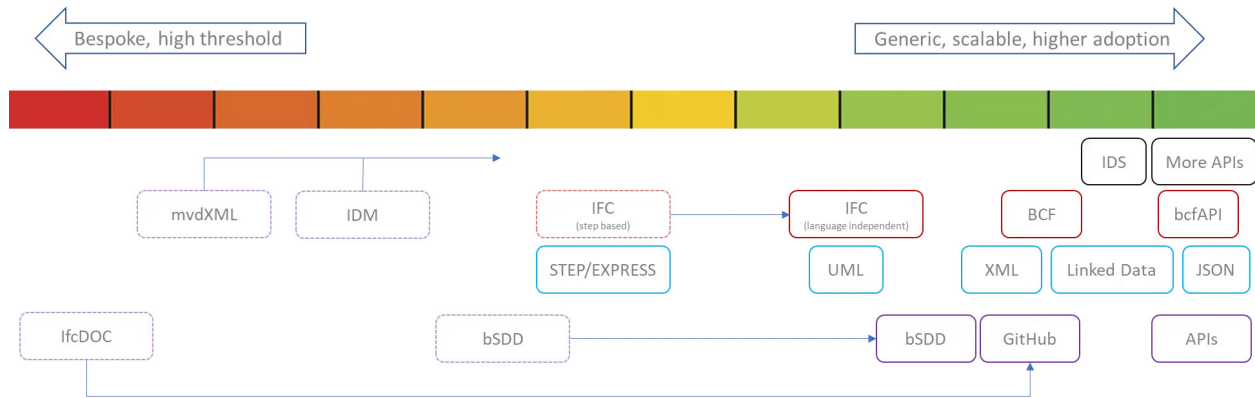


Figure 2-9: buildingSMART Technical Roadmap (buildingSMART 2020) [7]

range of tools then before [7]. According to their current roadmap, they aim to “optimize IFC to be used in transactional environments, instead of optimizing traditional file-based exchange” [7]. Thus, IFCJSON is currently being considered as a candidate for an official IFC data format as research and development progresses.

In addition to file formatting, the methods of how software platforms publish and digest information is an important consideration for information exchange. In modern development an Application Programming Interface (API) is a set of rules, protocols, and tools that allows different software applications to communicate with each other or other services. In simple terms, it serves as a bridge between software systems, enabling them to exchange information and interact with one another. APIs can be utilized to access the functionalities of a remote server or to use the features of one software in another, without having to know how the other application works.

One key concept in APIs is the idea of "state." In computer science, the state of a system refers to the current information maintained by that system at any given time. This information can include data, user inputs, and other relevant details required for the system to perform its functions. When it comes to APIs, a “stateful API” maintains information about all the interactions relating to the state, while a stateless API does not store any information about the interactions between communication moments. Statelessness is an

essential property for scalable systems, as it ensures that the server’s resources are not consumed by maintaining the state of individual clients.

Given this understanding of state, Representational State Transfer (REST) is an exchange style that defines a set of constraints for designing networked systems, with the goal of making it easier to integrate more services. REST is about organizing and structuring resources in a way that adheres to certain principles, such as statelessness and a uniform communication interface. This makes it easier to ensure that systems built with REST are easy to understand and can grow smoothly over time.

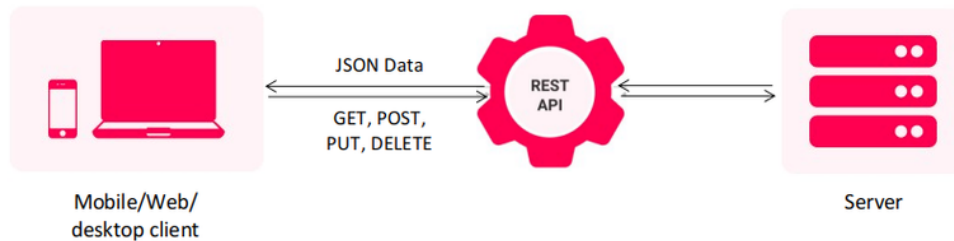


Figure 2-10: REST API Model (Saputra 2022) [41]

RESTful APIs leverage standard HTTP methods to perform operations on resources, which are usually represented by URLs. The most common HTTP methods used in RESTful APIs are GET (to retrieve a resource), POST (to create a new resource), PUT (to update an existing resource), and DELETE (to remove a resource). By using these standardized methods, RESTful APIs allow for a consistent and easy-to-understand way of interacting with web services. Each service in the system can be expected to use these four methods.

RESTful APIs, which adhere to the principles of the REST architecture, have become the preferred choice for many developers due to their simplicity, consistency, and ability to work seamlessly with web-based applications. However BIM software does not always have RESTful APIs. BIM can be considered stateful, as it maintains information about the current state of the building model, including the properties and relationships between its components. This stateful nature can make it more challenging for BIM software to

implement RESTful APIs, which are designed to work best with stateless systems. Moreover, BIM software often uses proprietary data formats and complex data structures, which may not be easily translated into the simple, uniform interface required by RESTful APIs.

The absence of RESTful APIs in some BIM software can affect its ability to exchange information with web services. It may require the use of custom APIs or data translation layers to facilitate communication between BIM software and other systems, which can be more difficult to develop and maintain. This can also result in reduced interoperability between different BIM software platforms and other applications, potentially limiting the effectiveness of information exchange. Additionally buildingSMART points out that the “.IFC” data format is challenged through RESTful API’s because of its legacy EXPRESS data structure. Therefore, it’s suggested that RESTful API’s are an important area of investigation for integrating web-based services into BIM.

While the overall patterning of a DT is likely the primary focus of my research, I have found that the data exchange considerations explored here are important factors to successfully integrating BIM and a web-based sensor network.

2.4 Findings from Precedents Research

In this background chapter I have first reviewed the origins and contexts of data formatting for both BIM and DT. What I found was that BIM was inherently developed as a contained, finite system for an individual to describe a building, while DT was developed as a technique to monitor dynamic systems. Now that both technologies have migrated to the web, there is a tendency to integrate the two, mostly as a response to managing a building’s real-time operations. However, utilizing a DT in the design or construction phase is under-explored.

Next, I reviewed existing DT prototypes researchers have built using BIM, in order to understand the integration between BIM models and real-time sensor networks to form a DT. What I found was that most research proposes extensive frameworks which either layer networks within BIM models, or connect BIM models to separate sensor networks. A gap

I identified both within the AEC industry and externally, is that there lacks a common architectural definition for DT.

Finally, I reviewed the state of the art for information exchange in web-based contexts. In response to BIM's increasing presence on the web, I reviewed the current research behind transitioning the IFC format into more progressive, web based structures. Much like the conclusions I drew about BIM's differences from DT, I found that IFC developers are bridging this data format gap by focusing on API development in addition to transitioning exchange formats to JSON.

Chapter 3

Hypothesis

3.1 Speculation

In an attempt to fulfill the overarching research question, "What is the software architecture of a building's digital twin?", my hypothesis aims to explore the potential advantages of using the SOA pattern, RESTful APIs, and JSON exchange format for integrating sensor networks with BIM software. The hypothesis posits that utilizing these approaches will lead to a more seamless integration of real-time sensor updates into a BIM platform, rather than using solely a monolithic software pattern and legacy IFC formats. This hypothesis keeps the research question at the forefront of its speculation and ultimately serves as an investigative tool for answering the research question.

In light of the findings from the literature review, I found it important to explore the formal structuring of a DT. The lack of a universally agreed-upon definition of software architecture for a DT is the first thing that was notable to me. While the International Organization for Standardization (ISO) acknowledges DT as a top emerging technology, it is still in the process of developing standards for a DT manufacturing network, including best practices for its technical architecture [18]. Nonetheless, research groups such as The Industrial IoT Consortium ¹ have proposed data communication standards, suggesting that

¹The Industrial IoT Consortium (IIC) is a global, member-driven organization dedicated to accelerating the growth and adoption of the Industrial Internet of Things (IIoT). It brings together industry experts, technology providers, researchers, and government organizations to develop best practices, guidelines, and frameworks for IIoT implementation. The IIC aims to foster collaboration, drive innovation, and promote

a DT should: have an information model with object types that can be browsed by other components have key-value data pairs which facilitate create, read, update, delete (CRUD) operations

Given the diversity and lack of formal structuring within research on integrating sensor networks with BIM, it is worth examining other architectural approaches that have proven successful in handling complex, heterogeneous systems [43]. One such approach is the Service Oriented Architecture (SOA), which has gained prominence alongside the IoT trend, where large-scale distributed systems of microservices on the web require architectures that scale on demand and evolve over time [24, 25]. I identified this approach because in most case studies reviewed, the sensor network consists of multiple, distributed sensor nodes, which each provide a single service and can be perceived as a microservice. Whether integrated directly into the BIM model, or connected to an auxiliary network, the sensors are consistently treated as a heterogeneous collection of individual components in existing research.

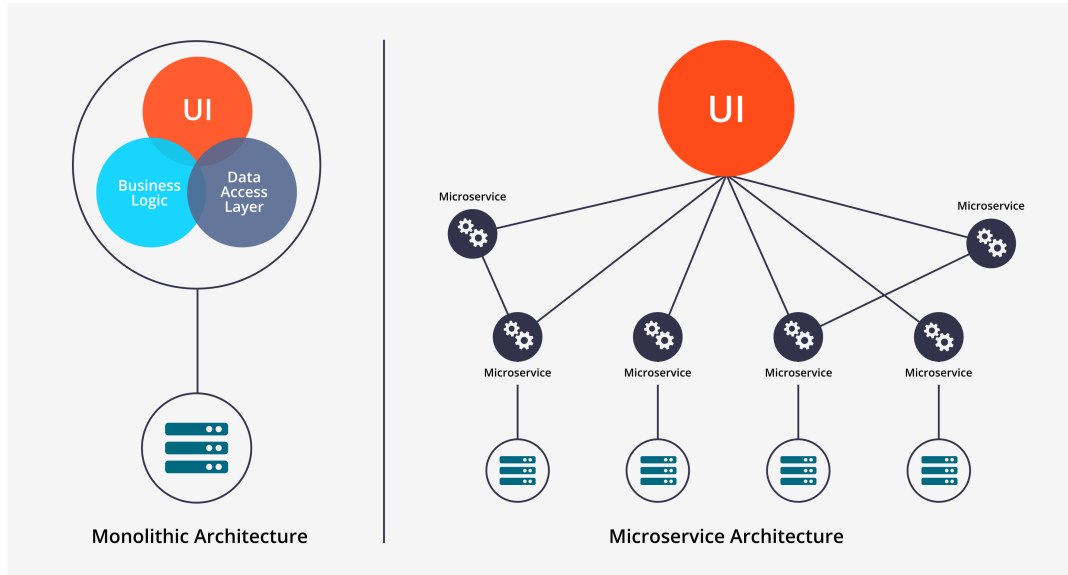


Figure 3-1: Monolith vs Microservice Pattern (Sanjaya 2020) [40]

Microservices offer a potential direction for further investigation, as they promote flex-
interoperability, security, and scalability within the industrial internet ecosystem.

ibility, maintainability, and resilience by dividing an application into a collection of single-function, self-contained, and independently deployable software components. This modular development allows individual services to be updated, deployed, or replaced without affecting the entire system. Microservices within an SOA have been well studied and successfully deployed, especially over the last ten years. Some of the most successful web companies such as Uber and Netflix run on microservice architecture. Given the scale and complexity of building projects, this software approach may also be a robust solution for structuring distributed sensor networks and data-rich information models. The microservices approach may provide valuable insights into how BIM and sensor network integration can be achieved more effectively, with better scalability and adaptability.

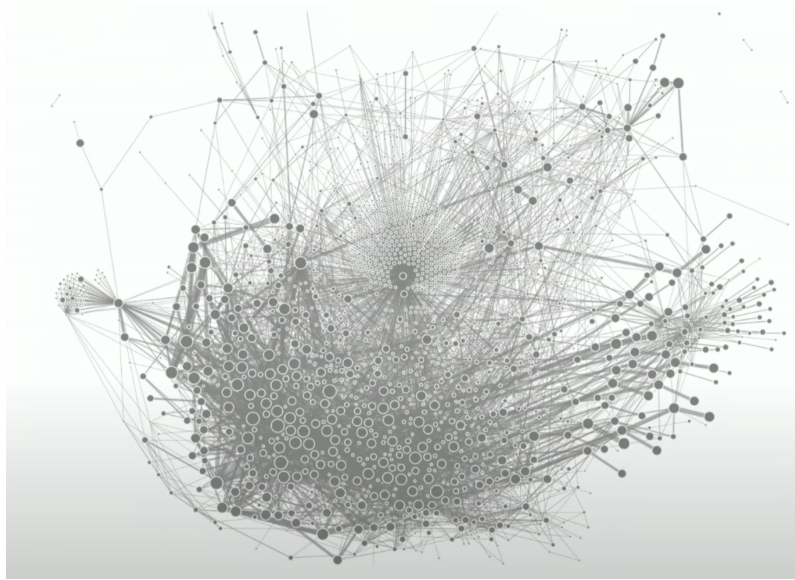


Figure 3-2: Uber's Microservice Architecture (Gud 2019) [17]

In the context of the AEC industry, microservices have been minimally researched for data exchange purposes, particularly during the building planning and construction phases [15]. By exploring the potential of microservices in the AEC domain, I may uncover new possibilities for integrating BIM and sensor networks, leading to the development of more efficient and flexible DTs that can enhance decision-making processes. This investigation

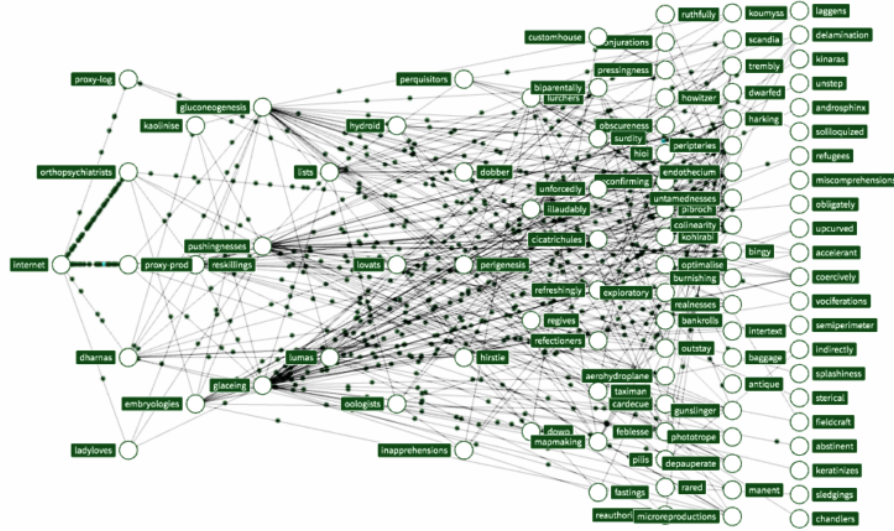


Figure 3-3: Netflix’s Microservice Architecture (Meshenberg 2016) [31]

could bridge the gap identified in the literature review and contribute to a better understanding of the relationship between BIM and sensor network integration in the formation of DTs.

While identifying a structural pattern for BIM and sensor network integration provides a possible template for the architecture of a building’s DT, the communication between objects has yet to be addressed. Distributing a system, as the SOA pattern suggests, relies on a greater amount of information exchange between the independent nodes. In the next section, I will address the information exchange as a component of the software architecture investigation.

3.2 Methods Overview

To support this hypothesis, I developed a prototype in the subsequent Methods chapter that may demonstrate the benefits of the proposed software architecture. This prototype will

not be an add-in² within a monolithic BIM application, but rather a collection of microservices designed to supplement existing BIM software. This approach contrasts with the more traditional add-in approach, which I speculate may be limited by the constraints and dependencies of a monolithic application. By extending the BIM platform with microservices, I hope to identify potentials in characteristics such as flexibility and interoperability between the various systems involved in the DT ecosystem.

Additionally, this prototype will utilize RESTful APIs and JSON data structuring. The use of RESTful APIs may also enable efficient communication between the BIM model and the sensor network, allowing for real-time updates and synchronization between the static model and dynamic components of the sensor network.

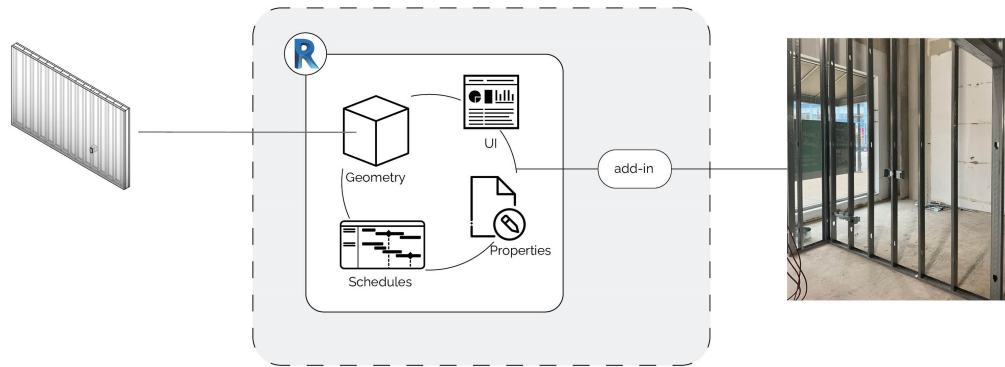


Figure 3-4: Diagrammatic Overview of Monolith-Oriented Digital Twin Prototype (Figure by author)

²An add-in for design software is a third-party software component that can be installed within an application to extend its capabilities and add new features or tools.

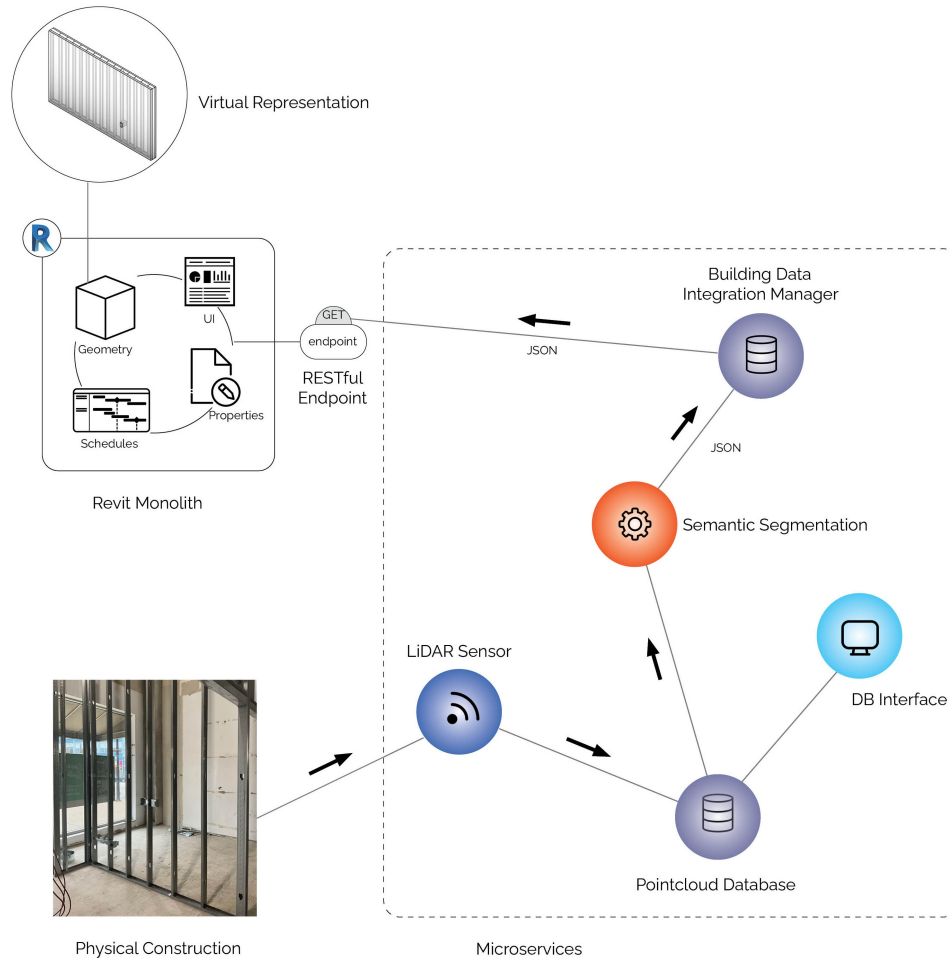


Figure 3-5: Diagrammatic Overview of Microservice-Oriented Digital Twin Prototype (Figure by author)

Chapter 4

Methods

4.1 Summary of Prototyping Technique

The intent of my prototype is to demonstrate the integration of a sensor with an existing BIM model to observe the potential advantages of utilizing an SOA pattern over the MVC pattern. To do this, I develop an application which takes raw data from a sensor and provides architectural information for the BIM model. The challenge I focus on is integrating this application into the BIM platform. I take two approaches: the first approach attempts to integrate this application directly within the standalone MVC-style software. Then, the limitations of this approach are identified. The second approach attempts to integrate the application into the BIM software following the SOA pattern, utilizing RESTful API's and JSON data formatting. The differences between the two approaches are then discussed.

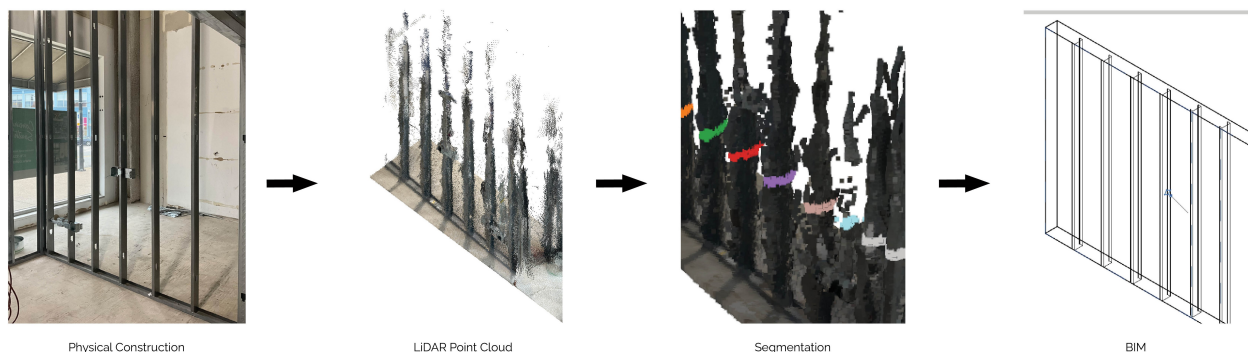


Figure 4-1: From Left to Right: Physical Wall, LiDAR scan, Stud Identification, BIM Representation. (Figure by author)

The workflow for both attempts will work as follows: a single LiDAR¹ sensor is used to update the design of a single architectural partition². Data is collected from the construction site at periodic instances with the sensor. Then, the core application is developed to automatically detect the number of structural studs³ installed, and determine their on-center spacing⁴. Finally, this data is translated back into the BIM model to update the virtual representation of the partition's structural system.

The decision to use a single LiDAR sensor on one architectural element was made to make the workflow as simple and straightforward as possible. The main focus will be observing the differences in structuring the software in each attempt. Additionally, I intend to document my process exhaustively to provide a comprehensive understanding of the components involved. Therefore, I begin with my data collection process in the subsequent section.

4.2 Construction Observation and Data Collection

The physical site for experimentation is the construction of an interior partition in an active construction site. I chose to study a partition because it is the most ubiquitous element in architectural design and construction, and could perhaps be the most relatable for future work. Additionally, partitions are so common that their anticipated construction timeline was highly predictable, allowing software development to happen in tandem.

To collect data, I visited an interior commercial renovation in Pittsburgh's South Side

¹LiDAR stands for "Light Detection and Ranging." It is a remote sensing technology that uses laser light to measure distances and create detailed 3D maps of objects and environments.

²A partition is another word for 'wall.' Partitions refer more specifically to the type of wall which separates space between two areas. This distinguishes it from other types of walls such as exterior walls which enclose conditioned space. Additionally, partitions are usually non-load bearing and are not weather barriers, making them much simpler in their design and performance requirements when compared to other walls.

³Studs are vertical structural elements in a partition which are installed in a repeated interval to resist lateral forces of a wall. They are the primary structural system in most partitions and can vary in their size, spacing, and material.

⁴On-center spacing is the distance between each repeated stud. The distance is measured from the centerline of the stud, to the centerline of the adjacent stud. Because structural studs are designed and installed at consistent, even intervals for structural continuity, this single dimension is commonly used to describe an array of structural members.

Works district, six times over a four month period to take photographs and interior LiDAR scans of partition construction activity. LiDAR was chosen as the sensor type for experimentation because it is a popular technology for capturing 3D objects, and is widely used on construction sites for 3D scanning. An iPad was used as a reasonably priced, off-the-shelf solution for photography and LiDAR scanning, that might be typically found on a modern construction site. I used the SiteScope iPad application, which utilizes the iPad's LiDAR sensor and camera simultaneously to create point clouds with RGB data, exported in the .ply format.

While many partitions were initially documented, a recurring challenge was poor lighting quality resulting in inaccurate point clouds. Therefore, a partition in a room against an exterior window was chosen as the best candidate for continued observation.

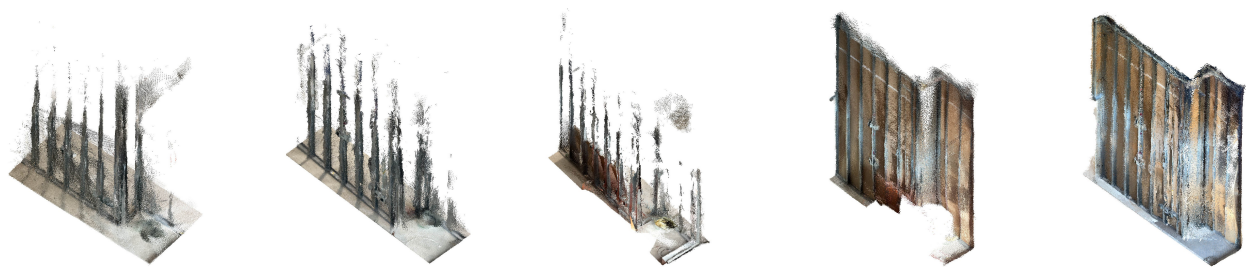


Figure 4-2: Periodic LiDAR Scans of a Partition Under Construction (Figure by author)

During the first visit, I observed that the cold formed steel studs we framed, and electrical junction boxes were set. During the second visit, I observed electrical conduit was dropped from the ceiling in between the framing, connecting to the junction boxes. Additionally, I noticed large traffic barriers stored in front of the wall, obstructing documentation. During the third visit, I observed drywall installed on one side of the framing (exterior of the room), and an increased amount of traffic barrier storage piled up against the wall. During the fourth visit, I observed no construction progress on the wall, but observed no storage occupying the space, allowing for full documentation. All of these conditions were challenges for parsing out and identifying individual studs, as each instance contained different amounts and types

of noise from one another.

4.3 Stud Detection Experiments

Once I had enough data, I began to develop the application which processes the raw point clouds and detects the studs. This is known as semantic segmentation, because it involves dividing the point cloud instance into segments based on unique objects.

in my first attempt to develop the application to detect structural studs in a point cloud, I performed segmentation by training a deep learning model on structural studs. I developed this work as a final project for a course titled “Machine Learning and Sensing,” in CMU’s Human-Computer interaction institute. This project seeks to use a pre-trained deep learning model to segment individual components in a wall which is under construction. in order to detect architectural features, a point cloud segmentation pipeline was developed based on the PointNet [49] deep learning architecture to convert point clouds from LiDAR scans into predefined categories. The categories for this experiment included structural studs, drywall sheathing, and plumbing pipes.

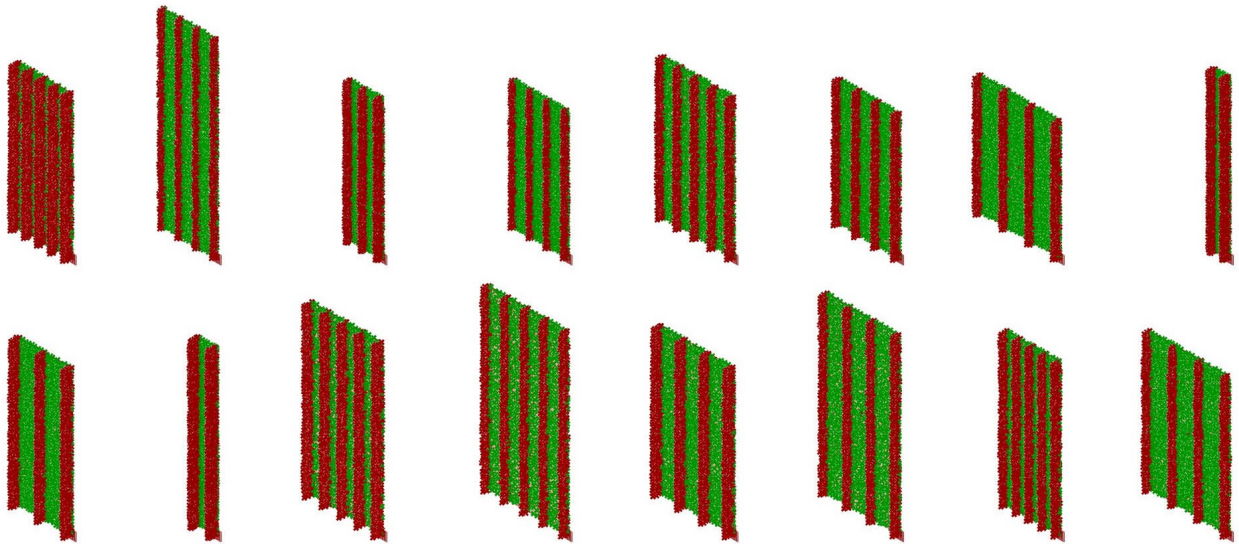


Figure 4-3: Subsample of Synthetically Generated Partition Dataset (Figure by author)

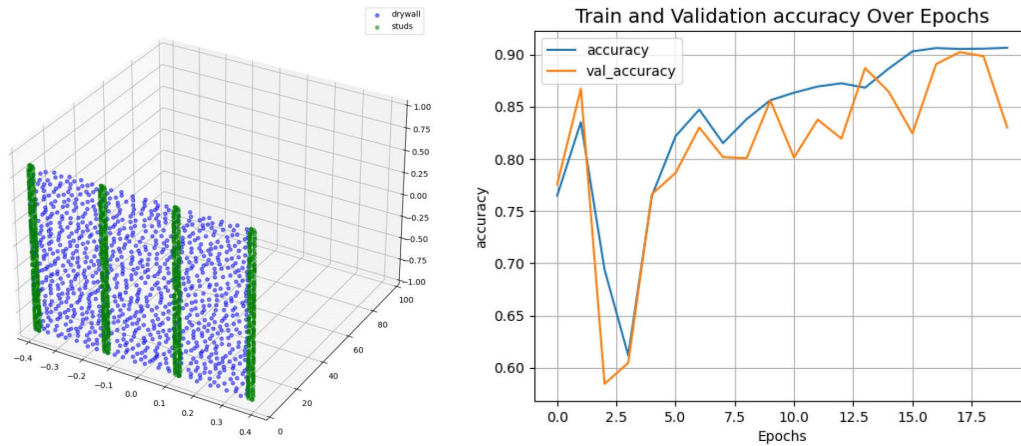


Figure 4-4: Left: Normalized Data Sample. Right: Train and Validation Accuracy Over Epochs (Figure by author)

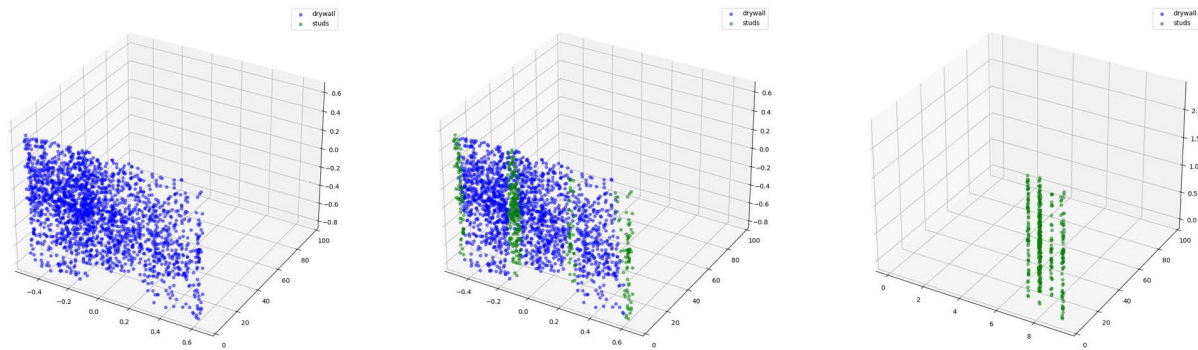


Figure 4-5: Evaluation Demo. From Left to Right: Raw Point Cloud, Stud Segmentation, Stud Isolation (Figure by author)

The PointNet model was trained with synthetic⁵ 3D instances of walls under construction converted to a point cloud representation. The data was generated with a Python⁶ script in Rhino.⁷ 4,000 wall instances were used, and the model achieved up to a 90% accuracy rate for detecting structural studs in the evaluation scans where drywall was present, but failed to detect any structural studs in evaluation scans where only studs were present. This was a major limitation for the LiDAR sensor I was using; it was unable to capture studs in full without a backdrop such as drywall. This is what led me away from training a learning model which might expect exact representations of studs, towards developing machine learning algorithms which could identify ‘stud-like’ geometry. In other words, the slender, repetition nature of the studs was unique enough from the surrounding elements, to where a rule-based style machine learning algorithm could be developed.

An alternative approach was explored for the detection and measurement of the wall’s structural studs in particularly imprecise point clouds. The next algorithm I developed extracts a slice of the point cloud at a specified height range to filter out ground noise. Then, a DBSCAN (Density-Based Spatial Clustering of Applications with Noise) algorithm is used to group the points within the slice into clusters as isolated stud instances. DBSCAN is a machine learning algorithm that groups together data points based on their proximity to each other. It works by identifying areas of high density and separates them from areas of low density to produce clusters of particular sizes. DBSCAN has two main parameters, epsilon and minimum samples, and it classifies each data point as a core point, border point, or noise point based on its density and the number of nearby points.

I used the DBSCAN implementation from the Open3D library as an efficient approach for

⁵Synthetic data refers to artificially generated data that mimics real-world data, and can be used to train deep learning models. By creating synthetic data, I could generate large amounts of labeled data quickly and efficiently, which can help to overcome the limitations of limited real-world datasets and improve the performance of deep learning models.

⁶Python is a high-level, interpreted programming language known for its simplicity, readability, and ease of use. It has a large and active community, making it a popular choice for a wide range of applications, including data analysis and artificial intelligence.

⁷Rhinoceros 3D, commonly known as Rhino, is a commercial 3D CAD software commonly used for architectural design prototyping.

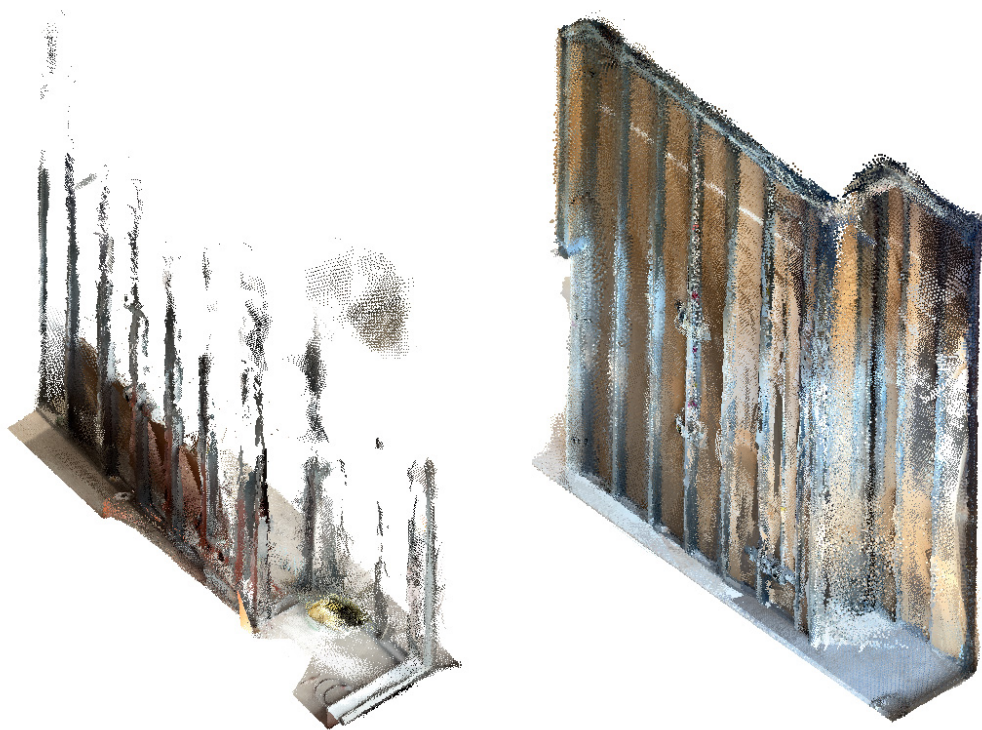


Figure 4-6: Left: Lidar Sensor Struggling To Capture Free Standing Studs. Right: Lidar Sensor Capturing Studs Against A Drywall Background. (Figure by author)

developing this prototype. Open3D is an open-source library that provides a comprehensive set of tools for processing and visualizing 3D data. Leveraging Open3D's implementation of DBSCAN ensured reliable and optimized performance, eliminating the need for engineering the complex algorithm myself.

The 2D XY center of each resulting cluster is computed and the distances between them are determined as the on-center spacing of the stud system. Additionally, the isolated clusters can be counted to provide quantity verification of physical studs in the wall. However, because the LiDAR scans tended to be noisy, the DBSCAN algorithm would often fail to isolate individual studs, as it would connect them with neighboring trails of noise.

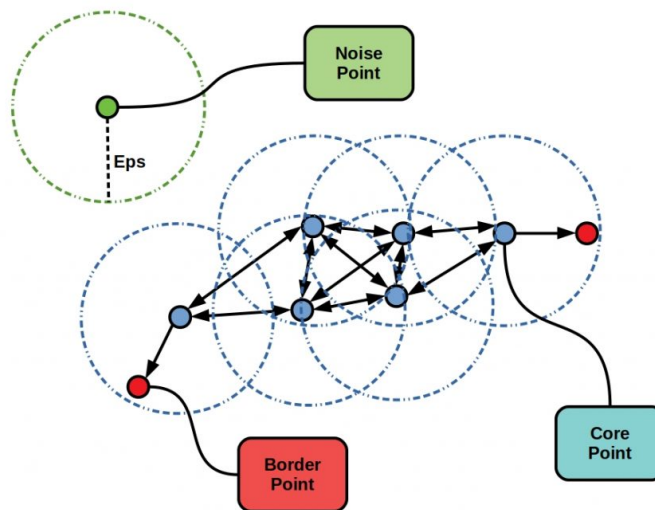


Figure 4-7: Dbscan Clustering Algorithm Mechanics (Rainerge 2020) [39]

A second version of the algorithm was developed with the OPTICS (Ordering Points To Identify the Clustering Structure) algorithm. OPTICS orders points based on their density and can identify clusters with varying densities, whereas DBSCAN is limited by its reliance on fixed density thresholds, which can lead to over-segmentation or under-segmentation of data points and difficulty in handling trails of noise. Additionally, I found OPTICS had a more robust ability to detect hierarchical cluster structures and can identify clusters of

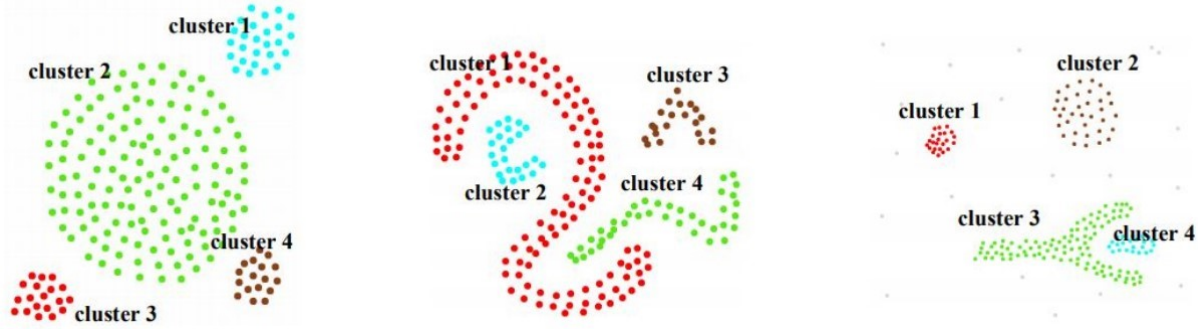


Figure 4-8: Dbscan Clustering, including Identifying Long Trails Of Points (Yeong 2019) [51]

varying densities allowing for more reliable segmentations of the studs in the noisy point clouds.

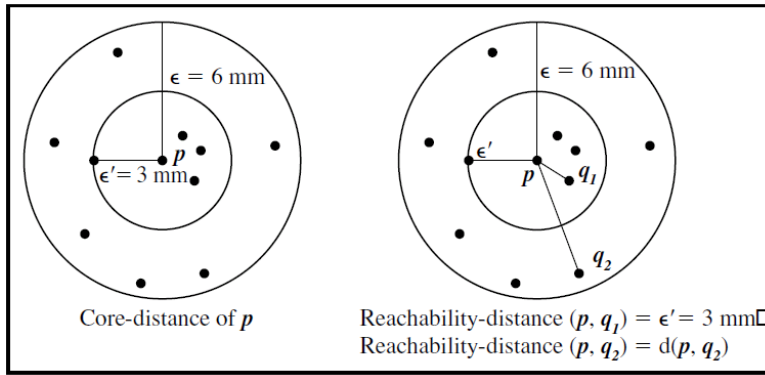


Figure 4-9: OPTICS Algorithm Mechanics (Anurag 2019) [2]

4.4 MVC Add-in Implementation

With the working point cloud segmentation application, the next step was to develop it as an add-in to a pre-existing MVC-style BIM software. The add-in was developed for the Revit software platform. It was designed to be integrated directly into Revit's MVC architecture; meaning it accesses Revit's data directly (the model), uses only Revit's GUI (the view), and is developed with Revit's existing API (the controller). The add-in focuses on providing the

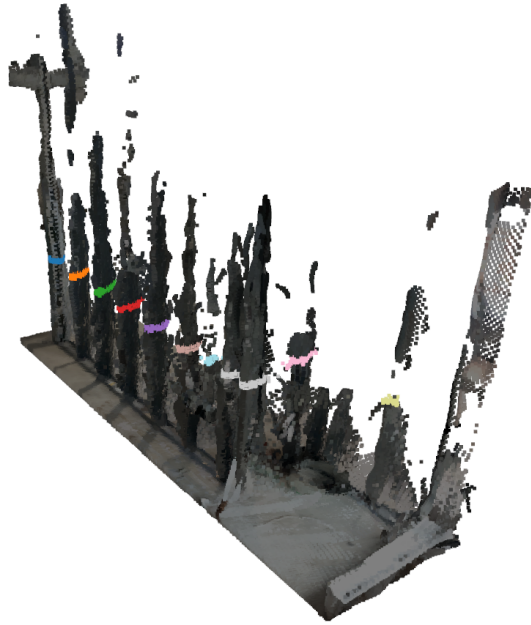


Figure 4-10: Visualization Of Optics Implementation Showing Stud Identification and Quantification (Figure by author)

same point cloud segmentation service described in the previous section.

The decision to develop the add-in using Python was driven by the extensive support for data science and open-source machine learning libraries available in the language, which became essential for the point cloud segmentation application. The primary problem was that Python is not a programming language that works well with Revit's .NET framework.

The .NET framework, upon which Revit is built, is designed to natively support languages like C and C++, rather than Python. These native languages are more focused on application development and do not offer the same level of support for data science and machine learning libraries as Python does. This discrepancy arises because the .NET framework was created by Microsoft to cater to a wide range of applications and services, prioritizing code reusability and security. Consequently, the framework is optimized for languages like C and C++, making Python integration non-native and more challenging.

Languages native to the .NET framework, such as C and C++, do not offer the same

level of support for data science libraries and are primarily used for application development rather than machine learning tasks.

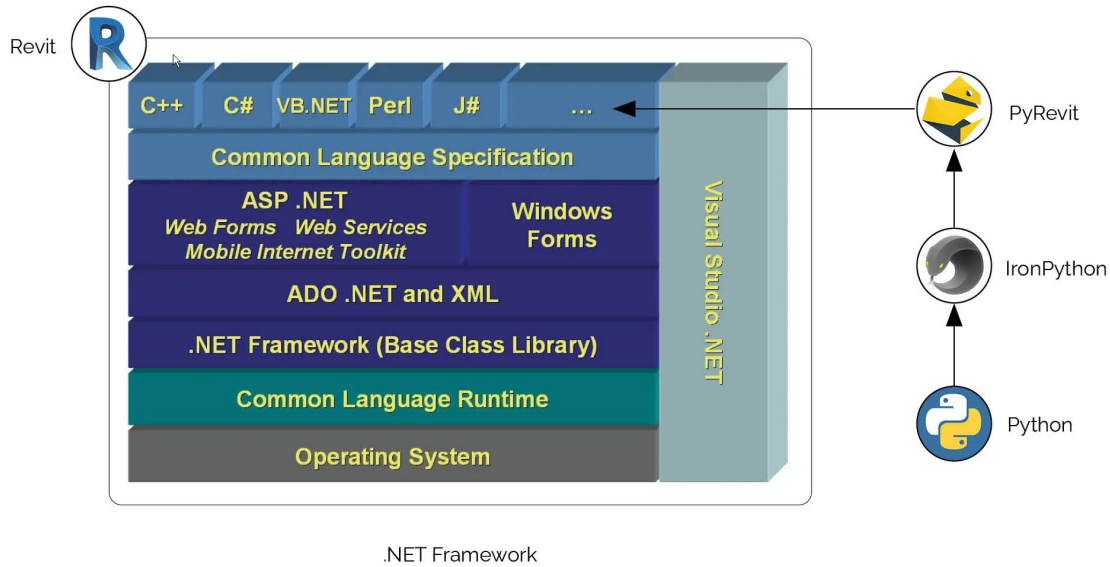


Figure 4-11: Diagram Showing The Chain Of integration For Python in Revit's .Net Framework (Lopes and Eracar 2011) [28]

Therefore, PyRevit was utilized to bridge this gap and enable the integration of Python into the .NET framework. PyRevit relies on IronPython, a .NET implementation of the Python language, to achieve this integration.

However, during the development process, it was discovered that IronPython does not support external libraries, which are essential for machine learning tasks. To address this limitation, an attempt was made to use CPython in PyRevit. PyRevit has limited capability to run on CPython and supports only one version of CPython, which is 3.8.5.

When installing libraries in the CPython⁸ environment, it was found that NumPy⁹ worked

⁸CPython is the reference implementation of the Python programming language, written in C, that provides an interpreter for running Python code. PyRevit is not designed to run on CPython because CPython does not provide the same level of compatibility with the .NET ecosystem as IronPython, which can limit its ability to interact with .NET-based applications like Revit.

⁹NumPy (Numerical Python) is a Python library that provides support for large, multi-dimensional arrays and matrices. It is particularly useful for optimizing the enumeration and calculation of large data sets such as point clouds.

and was capable of performing tasks such as counting points in the point cloud. Open3D also worked, enabling the use of its built-in algorithms, such as DBSCAN, to perform calculations. However, when importing the Scikit-learn library to use the OPTICS algorithm, it had overwritten all other libraries it depended on, causing the individual libraries which initially worked to become dysfunctional. Consequently, the PyRevit environment attempting to run the segmentation application encountered dependency management complexity.¹⁰ in this situation, using an environment like Conda¹¹ or Node.js¹² could potentially help to automatically manage library dependencies. Unfortunately, PyRevit, specifically tailored to Revit, is not compatible with Conda nor Node environments. Thus, the user is left to manually manage dependencies for the add-in application, where finding a workable solution is difficult if not impossible.

The key takeaway from this development process was that creating machine learning-dependent add-ins like the point cloud segmenter for structural studs within Revit’s MVC architecture is highly challenging due to the heavy reliance on data science libraries. These libraries require additional tools for dependency management, which need to exist outside of Revit’s monolithic environment. The complexity encountered during the development of the add-in highlights the limitations of integrating advanced machine learning-style analytic capabilities into BIM applications and emphasizes the need for alternative approaches, such as the proposed microservices-based architecture.

¹⁰Dependency management complexity refers to the challenges that arise when managing software dependencies, including the need to ensure that all required dependencies are installed and compatible with each other. With numerous dependencies, there is a potential for conflicts or errors when updates or changes are made to these dependencies. As software projects become more complex and rely on a larger number of dependencies, managing these dependencies becomes increasingly difficult and can lead to issues with compatibility.

¹¹Conda is a package management system for installing, running, and updating software packages and their dependencies. It allows users to easily manage different versions of software packages and create isolated environments for different projects or workflows.

¹²Node.js provides a package manager called npm (Node Package Manager), which allows developers to easily manage libraries and dependencies.

4.5 Microservice Implementation

in contrast to the add-in implementation described in the previous section, this section details the development of a small collection of microservices designed to integrate the point cloud segmentation application while addressing the limitations associated with the add-in approach. The proposed microservices-based architecture leverages Docker containers, which offer a best-practice solution for building and deploying microservices. Docker containers excel at managing imports, dependencies, and data science libraries, effectively addressing the dependency management complexity encountered in the previous attempt.

Docker containers are designed to run on any operating system, which is already fundamentally more flexible than Revit’s .NET framework from Microsoft. To ‘launch’ a Docker container, developers focus on building instructions (specifying the libraries to use, the programming languages to use, etc) called Docker images. Docker images serve as instructional templates to ensure a successful microservice build each time they are launched to the web, making them highly reliable, and able to detect errors more frequently.

The first step in developing the collection of microservices for the prototype was to create a point cloud database microservice to store raw point cloud images. Unlike the add-in which utilized Revit’s database, the microservice prototype needed to build all of its own services “from the ground up,” because they operate independently from the BIM software. The point cloud database was built using a service called MongoDB. MongoDB is a NoSQL¹³ database management system that provides a fast and flexible database platform, making it a good choice for creating the prototypical database. Continuing this “ground-up” approach, a user interface was developed next to interact with the database.

Next, the database interface microservice is built using PyMongo, Streamlit, and Pandas. PyMongo is a library that facilitates interaction with MongoDB using Python, while Streamlit is a web framework that allows users to create custom web applications that can

¹³NoSQL databases such as Mongo are typically columnar databases, as opposed to traditional SQL databases, which are row and column which exhibit slower data indexing and retrieval performance [50].

be run on most web browsers. Pandas is a powerful data manipulation library, which is particularly useful for formatting and displaying data in easy to read tables. These libraries enabled the development of a user interface that allows users to:

1. Upload point cloud files from an iPad.
2. View a table of all the point clouds in the database, including their file type, size, and number of points.
3. Select files to download to a local drive.
4. Clear all files from the database if needed.

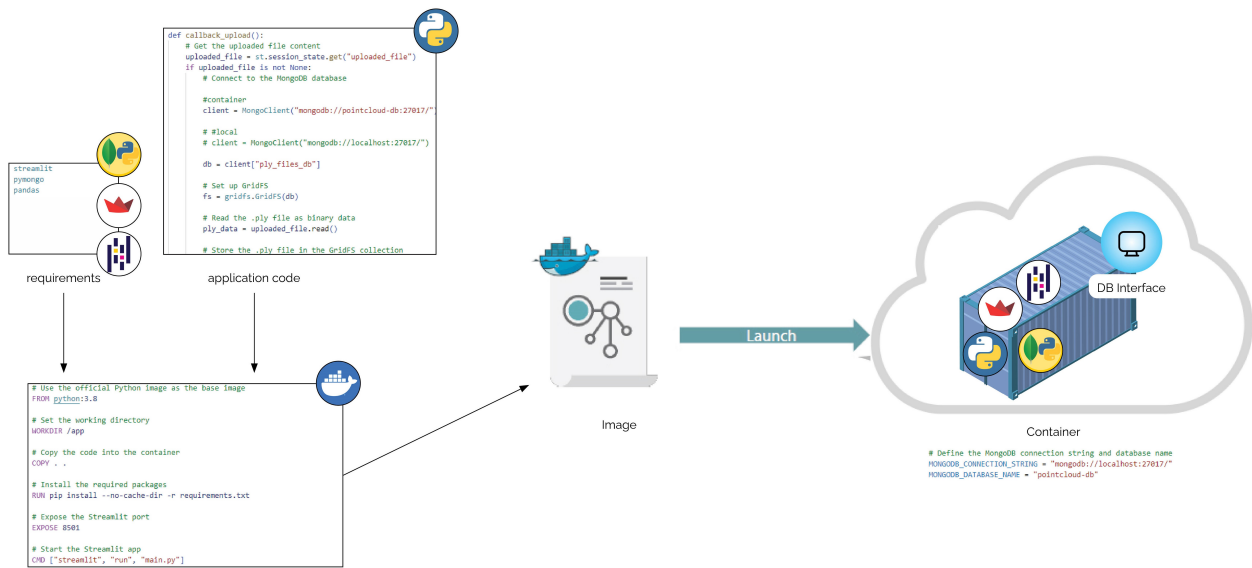


Figure 4-12: Diagram Showing The Construction Of The Docker Container For The Database interface Microservice (Figure by author)

Following the development of the database interface, the point cloud segmentation application is created as a microservice. As highlighted previously, this microservice relies on several libraries, including Open3D, Plyfile, NumPy, Scikit-learn, SciPy, PyMongo, Matplotlib, Streamlit, and Pandas. This approach demonstrates the effective handling of extensive package management within an independent microservice, despite some overlapping

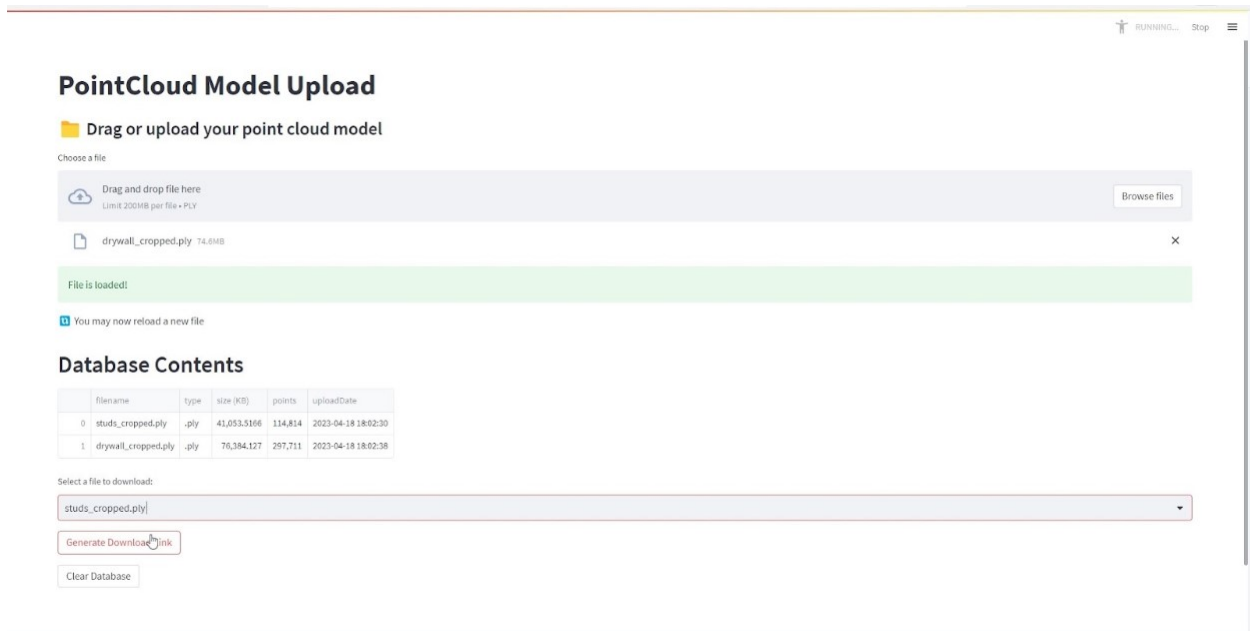


Figure 4-13: Database interface Microservice Deployed and Running in The Chrome Web Browser (Figure by author)

dependencies with other microservices, such as Pandas and Streamlit found in the database interface microservice.

Next, a service I titled the ‘BIM Data integration Manager’ (BDIM) microservice is developed to store the resulting data from the point cloud segmentation service in a JSON data format. The BDIM’s purpose is to integrate information from any data processing services, such as the stud detector, and relay the results to any BIM models which subscribe to it. Utilizing a software-agnostic data structure (JSON) and provides updates via HTTP¹⁴ requests through a RESTful API. The BDIM uses Flask¹⁵ to create a HTTP endpoint for this type of communication. From a more abstract perspective, the BDIM stores information

¹⁴HTTP (Hypertext Transfer Protocol) is a widely used protocol for communication on the internet, allowing data to be transferred between web servers and applications. in the context of the BDIM microservice, HTTP requests are used to send and receive data between the BDIM and other BIM models or data processing services.

¹⁵Flask is a popular Python web framework that is used to create web applications and RESTful APIs. It is lightweight and easy to use, making it a popular choice for prototyping and small application development. in the context of the BDIM microservice, Flask is used to create an HTTP endpoint that can receive and respond to requests from other applications or services.

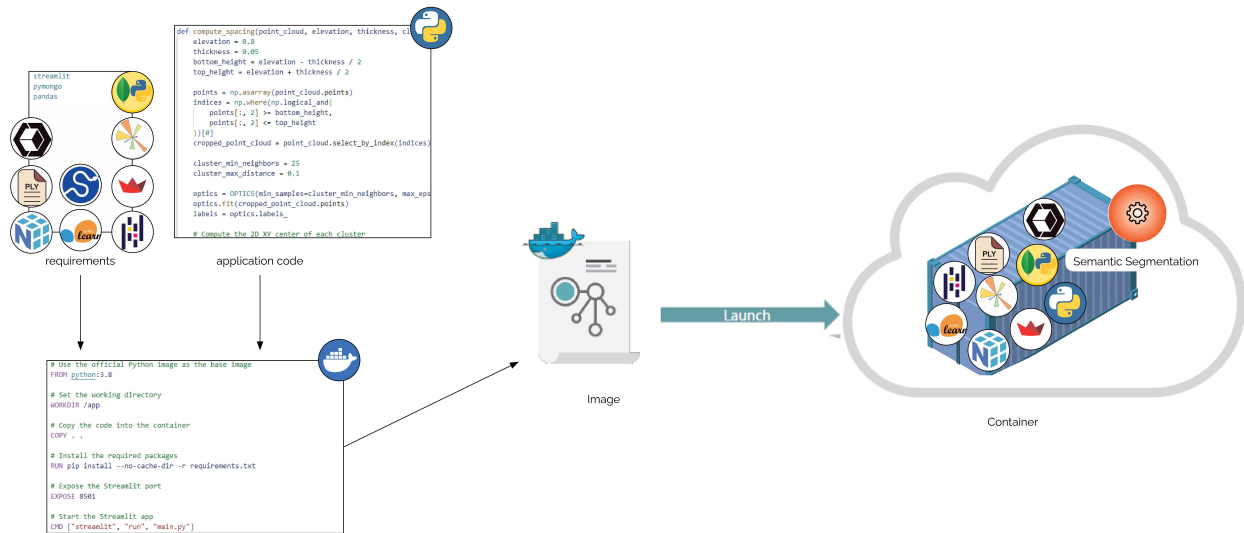


Figure 4-14: Diagram Showing The Construction Of The Docker Container For The Stud Spacing Analyzer Microservice (Figure by author)

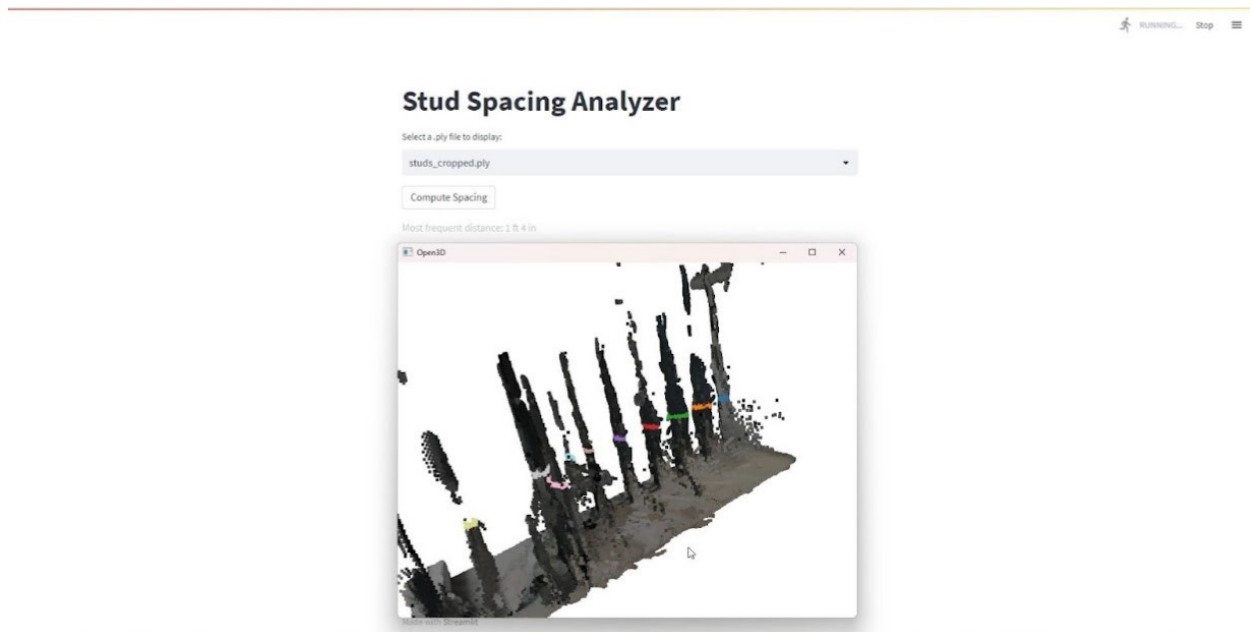


Figure 4-15: Stud Spacing Analyzer Microservice Deployed and Running in The Web Browser (Figure by author)

about the studs and allows any BIM software to retrieve the data with one, standardized request.

Finally, to subscribe to the BDIM, a simple HTTP endpoint is built into Revit as an add-in. Unlike the previous attempts with PyRevit, this add-in only requires one library, Flask, to build and expose the endpoint for retrieving stud spacing information to update the BIM model. In Revit, users can click 'update,' and if updated data is available in the BDIM, the partition's structural system will update accordingly. From a more abstract perspective, this add-in relies on the microservice network to perform all of the complex data handling and point cloud computations, and simply subscribes to the results once they are available. It quite literally offloads the application complexities into other independent web-hosted applications.

In conclusion, the microservices-based approach for integrating the point cloud segmentation application into the BIM platform demonstrates a successful alternative to the add-in implementation within Revit's framework. The methodology adopted in this section, which entailed building a collection of microservices, leverages the advantages of Docker containers and web-based communication, effectively addressing the limitations and dependency management issues encountered in the add-in attempt. This approach showcases the potential of a distributed and modular architecture to facilitate the seamless integration of diverse data tasks for BIM platforms.

While this chapter provides an overview of the experimental process and outcomes, the following chapter will delve deeper into the comparisons between the add-in and microservices-based approaches, thoroughly examining the advantages and disadvantages of each solution.

Chapter 5

Observations

5.1 Findings

This chapter expounds on the proposed and tested development of the point cloud segmentation application, both as an add-in and as a collection of microservices, comparing the two approaches in terms of their efficacy and limitations. The ensuing observations elucidate the distinction between the two implementations and underscore how the microservice architecture may offer a more seamless integration of real-time sensor updates for BIM platforms.

The initial observation pertains to dependency management and libraries. The development of the add-in encountered a primary challenge encountered in the form of the complexity of dependency management. The segmentation application that was intended to be integrated was dependent on extensive machine learning technology. Integrating machine learning capabilities such as point cloud segmentation into Revit's MVC architecture proved to be challenging to the reliance on data science libraries that require additional tools for dependency management. The add-in approach, which was developed in PyRevit, encountered limitations initially with IronPython's support for external libraries, followed by compatibility issues while attempting to use specific CPython versions.

It is worth noting that PyRevit, being a third-party application developed independently from the developers of Revit, supports only one version of CPython. This limitation arises primarily because PyRevit is the work of a single developer, who has complete discretion over the deployment and accessibility of CPython versions, both currently and in the future.

Despite being open source, this aspect of PyRevit’s development poses a potential risk in terms of its long-term sustainability.

In contrast, the microservices-based approach presents an effective solution to the issue of dependency management through the use of Docker containers and the distribution of services into service-oriented containers. By dividing the application into smaller, independent services, each with its distinct set of dependencies, the overall complexity of dependency management is reduced. Docker containers manage imports, dependencies, and data science libraries while ensuring version control, offering a more reliable solution. Moreover, Docker containers are ‘built’ every time they are deployed, facilitating the identification of outdated libraries or broken dependencies.

The microservices architecture allows for the use of various libraries, including those with overlapping dependencies, without compromising the functionality of individual services. This is feasible as each service functions within its own isolated environment, which mitigates clashes between dependencies. In the Revit environment, all functionalities are anticipated to operate within a single environment.

In other words, distributing the services into service-oriented containers provides a more efficient means of managing dependencies, as opposed to managing all dependencies within a single environment. This approach allows for easier updates and upgrades, as developers can update individual services without disrupting the entire application. Furthermore, the distributed nature of microservices may improve fault tolerance, as a failure in one service does not necessarily cause the entire system to fail.

The second observation pertains to the issue of scalability and flexibility. The monolithic structure of the add-in implementation within Revit’s framework renders it less scalable and flexible for seamless integration with sensor networks. This is because it relies on a single environment for managing dependencies. For instance, in the context of the point cloud segmentation application, if the user wants to switch between the OPTICS and DBSCAN algorithms for segmenting point clouds, they would have to contend with different library

dependencies, which would create compatibility and functionality issues. Furthermore, the add-in approach does not accommodate the dynamic nature of DTs, as it is tightly coupled with the BIM software, rendering real-time updates and integration with external services challenging.

In contrast, the microservices architecture provides a more scalable and flexible solution, by providing independent services that can separately be updated, maintained, and scaled. The point cloud segmentation application built as a microservice demonstrates that it can smoothly switch between the OPTICS and DBSCAN algorithms without affecting other services. Each microservice operates in its isolated environment within the system. This level of independence enables seamless integration of diverse data sources such as sensor networks, with minimal impact on the overall system.

The final observation pertains to the use of RESTful APIs and JSON data format, which promote efficient communication between the BIM model, sensor networks, and the various associated microservices. This use facilitates real-time updates and synchronization of static and dynamic components of a DT. Specifically, any modifications to the point cloud segmentation application can be conveyed to the BIM Data Integration Manager (BDIM), and any HTTP-enabled BIM software can subscribe to the BDIM to receive the updates. This method streamlines the integration process, allowing BIM platforms to accommodate new technologies or services and adapt to changing demands.

5.2 Limitations

Although the microservices architecture offers advantages over the add-in approach, it has some limitations worth noting. The decision to employ microservice technology and SOA pattern was based on its established use case and proven utilization. However, creating basic prototypes using this pattern overlooks several critical features that the SOA pattern depends on.

The first limitation is data consistency; the use of separate databases in microservices can

present challenges to maintaining consistency in data. This can be particularly problematic in distributed systems where multiple services need to update the same data. In the presented prototype, it would be desirable to ensure that once a point cloud has been processed and relayed to the BIM model, the original point cloud database documents that the task has been completed to avoid duplication of work. While manual operation of the microservices can be sufficient for simple experimentation, the SOA would require automatic synchronization to function as intended . Therefore, developers must thoughtfully design and implement data management strategies to uphold data consistency.

The second limitation concerns security. When multiple services communicate over a network, there is an increased risk of security vulnerabilities. Although microservices operate independently, each service must be secured individually. In applications operating at the scale and complexity of a construction site, the communication between services may require to encryption and authentication to prevent unauthorized access or data breaches. Implementing and maintaining robust security measures across all microservices can be a daunting task

The third limitation concerns the increased software architecture complexity. While the prototype explored only utilized a single sensor to integrate updates into the BIM model, a sensor network, as discussed in the background chapter, may employ multiple sensors, each with their unique data types and interfaces. Furthermore, exploiting the advantages of microservices architecture can result in a large number of independent services, which may significantly increase the overall orchestration of the system beyond the scope of monitoring the structure of one partition. Buildings are inherently complex, and one DT may have hundreds of services to support sensor network integration. Consequently, developers may need to manage and maintain these services, each with its own set of dependencies, configurations, and communication protocols. Such management and maintenance can lead to increased overhead, such as steeper learning curves and higher development and operational costs.

Chapter 6

Conclusion

6.1 Discussion

In this thesis, an investigation was undertaken to explore the challenges associated with integrating real-time sensor data and incorporating advanced machine learning capabilities into conventional monolithic BIM software architecture. The limitations of the conventional BIM technology were identified through the development of a semantic segmentation application as an add-in for Revit’s monolithic MVC architecture. To address these challenges, a microservices-based approach was proposed to facilitate the seamless integration of computationally demanding tasks for BIM platforms. In this chapter, I shall reflect on the proposed approach and discuss potential impact they may have on the research direction of DT software development in the AEC industry.

The research offers a potential avenue towards redefining the conventional approach to BIM data management, exchanges, and flows within the AEC industry. By deviating from the conventional monolithic structure of BIM as advocated by Eastman’s BDS, I have delved into a new distributed system characterized by a network of task-oriented services. This innovative approach to architectural information integration prioritizes the establishment of a distributed system of computations that can aptly address the increasing complexity of the AEC industry.

By drawing parallels to the evolution of software engineering, this approach to BIM data management aims to break down the overall BIM space into smaller, more manageable

components, in a manner similar to the shift in software development towards embracing extensive ecosystems of libraries and dependencies. By prioritizing a more distributed system, this approach seeks to encourage greater participation among stakeholders, promoting more effective management and collaboration, and creating a more inclusive development environment in the AEC industry. This innovative perspective on architectural information integration builds upon the motivations presented in the introduction chapter, advocating for a more effective industry that is fostered by enhanced collaboration.

The overarching objective of this research is to facilitate an industry that is more decentralized and diverse, with a software architecture that accommodates these traits. Drawing inspiration from Eastman’s object-oriented data representation as a visionary approach that advanced the representation of architecture in the context of data structuring, I put forward a comparable vision that responds to the changing nature of software development in the present-day era. By adopting the collaborative and dependency-driven approach of modern software engineering, I aspire to foster a more interwoven and innovative AEC industry.

Through the adoption of a distributed approach, this research opens up new avenues for future exploration of the social and participatory aspects of the AEC industry. The development of this new, segmented space allows for a more open-sourced environment, facilitating a wider range of participants to contribute to the design, construction, and management of buildings. This visionary perspective demands a rethinking of the way we approach BIM data management and exchange and reimagining a more collaborative and interconnected industry.

The benefits of this vision could be characterized though hypothetical, yet common situations in AEC practice. For example, one of the difficulties during the building planning and construction process is keeping track of project costs. At the core of the issue, material and labor costs fluctuate based on market demand, and subcontractors are the project stakeholders closest to understanding current pricing data. However, the project designers (architects and engineers) do not typically work directly with subcontractors, as their chain of commu-

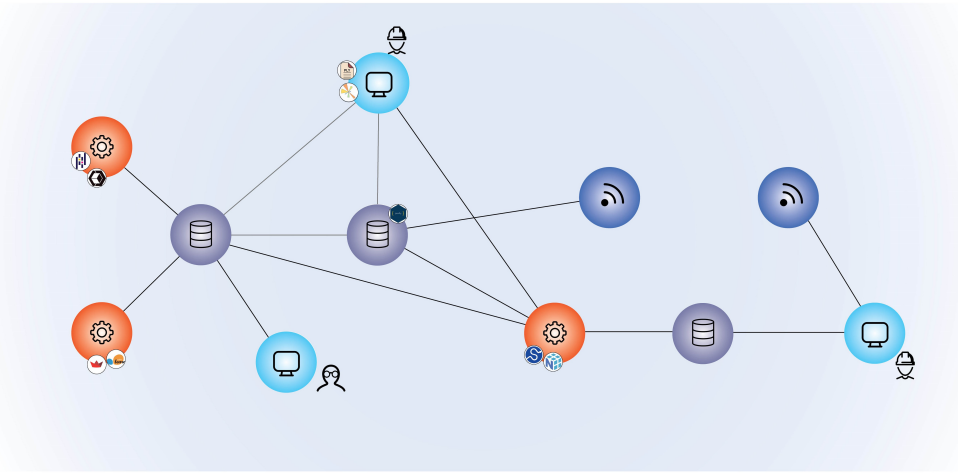


Figure 6-1: Vision For A Task-Oriented Ecosystem. (Figure by author)

nication will go through the project owner, then to the general contractor, and finally to the sub-contractors. Therefore, if a designer is tasked with meeting an overall budget, their gauging of material costs is left to either generalized estimations based on historical data, or lengthy communication channels which create slow feedback loops. Due to the loosely accurate estimations, project budgets are commonly exceeded or underutilized by the time the project is complete.

In a possible scenario within a distributed network approach, a subcontractor's pricing data could be treated as an online node of real-time, fluctuating information for materials and labor of their specific trade. Then, a designer can subscribe their model to this data point and get real-time feedback on total project costs based on the model's contents. Rather than estimating a project at major milestones (typically months apart), the project budget can be tracked, and responded to, in real time as both design criteria and market conditions change. In addition to agility and accuracy, as the project grows in complexity, additional subcontractors' pricing data could be integrated into the network. The DT may begin with a few significant pricing nodes such as steel and glass, but towards the end of the process may have hundreds of pricing nodes including furniture, fixtures, and equipment. Additionally, with a commonly enforced protocol, pricing nodes may be swapped out easily, allowing for a

seamless comparison between steel and wood costs at any time, for example. Furthermore, the real-time tracking of pricing could enable valuable data insights towards unforeseen trends. For example, escalation¹ is sometimes an overlooked issue that can impact project budgets. By understanding timelines of price creep, better statistical models can be built for improved predictability of future work. Finally, easily integrating pricing nodes in a DT could foster a more inclusive price bidding process for projects. Project managers may be able to evaluate many bids at once, rather than only a handful, and the process for contractor bidding may be as easy as subscribing their data to the DT. By lowering the overhead for everyone involved, the bidding process could be more open and efficient.

The aforementioned scenario demonstrates more direct information sharing between previously distant collaborators, enabling a more efficient and seamless cost management process. In addition, it may allow for a more inclusive platform for subcontractors to participate in the pre-construction process. This narration highlights improved collaboration through increased contribution from stakeholders. Another possible scenario could highlight improved collaboration through easier access to data to stakeholders who may have previously not interacted with design models

In this second scenario, perhaps involving a data-rich DT of a large mixed-use development, the diversity of project stakeholders necessitates distinct modes of interaction with the DT. Here, a microservices-based DT offers tailored user interfaces for each stakeholder, all accessing the same underlying DT. For instance, the project manager, familiar with spreadsheets, can interact with a structured data interface for easy tracking and resource management. The architect, needing a more visual approach, can utilize a 3D modeling interface allowing detailed design exploration, both interfaces optimized for varying computational capabilities of their respective devices.

Conversely, the marketing team can engage with a photorealistic, immersive VR² envi-

¹Escalation refers to the increase in cost of labor and materials over time due to various factors such as inflation, market demand, and supply chain disruptions.

²Virtual Reality (VR) refers to an immersive digital simulation of a 3D environment that can be experi-

ronment for compelling project promotion, run on high-performance VR devices. On-site teams, potentially less technologically dependent or without high-end hardware access, are equipped with a simplified, touchscreen-friendly interface on tablets and smartphones. Each of these interfaces, generated by dedicated microservices, ensures all stakeholders can contribute to and utilize the same DT. This approach not only enhances collaboration but also democratizes access, fostering an inclusive and participatory AEC industry.

6.2 Contributions

This research has the potential to make several contributions to the field of BIM research and the wider DT research conversation in the AEC industry. By exploring new approaches to BIM data management and sensor integration, the findings of this thesis might pave the way for more efficient and flexible solutions that better accommodate the diverse and often fragmented nature of the industry. Several contributions have been identified as follows:

The first is the investigation of integrating sensor data into Revit. This research identifies and analyzes the limitations of Revit’s software architecture, specifically in the context of integrating real-time sensor data and leveraging machine learning capabilities. By examining the challenges faced when attempting to develop a semantic segmentation application as an add-in, this work highlights the need to explore alternative approaches that can address these limitations.

The second contribution involves the conceptual prototyping and implementation of a microservices-based DT. In response to the identified challenges, the proposed architecture for integrating sensor updates into a BIM platform utilizes microservices. The implementation of a collection of microservices provides evidence of the potential advantages of this approach, such as improved dependency management, more efficient communication, and the ability to accommodate diverse data formats, languages, and libraries.

The third contribution pertains to a visionary perspective on the future of architectural

enced through a headset or other devices.

data management, proposing a distributed, task-oriented BIM system that challenges the traditional monolithic BDS-inspired paradigm. This perspective draws from the successful implementation of the microservices-based solution, which serves as a promising foundation for a more inclusive and collaborative approach to BIM. The proposed system is designed to encourage efficient and cooperative practices within the AEC industry, offering new avenues for exploration and potential for transformative change.

The fourth contribution entails the demonstration of using the JSON exchange format and RESTful APIs for architectural data exchange. This research showcases these technologies which align with buildingSMART's current roadmap. By adopting these approaches, the BIM data exchange process (particularly in the BDIM microservice) became more efficient and web-friendly. As such, the research presented herein serves as a working example of the forward-thinking concepts espoused by buildingSMART.

The fifth contribution presents a critique of PyRevit's limitations in integrating data-science capabilities. Through an examination of the challenges encountered in developing a point cloud segmentation application as a Revit add-in, this research highlights the limitations of incorporating advanced data-science capabilities into BIM applications using PyRevit. The findings underscore the difficulties in managing library dependencies and the incompatibility of certain environments, such as Conda or Node.js, with PyRevit. This critique showcases the importance of exploring alternative approaches to harness data-science and machine learning capabilities in the context of AEC use cases.

6.3 Next Steps

Expanding upon the research and findings presented in this thesis, there are several future directions to further advance and refine the proposed microservices-based approach for sensor network integrations in BIM. Firstly, the three issues identified in "Limitations" section could serve as immediate areas for further development. Additionally the following topics could be explored for future work.

The first topic for exploration to broaden the prototype to incorporate a broader range of sensor types and interfaces. Although this research focused on one sensor and addressed only the point cloud data for translation, wider IoT and sensor network ecosystems suggest that many sensors and data types, including photographs, occupancy sensors, or RFID tags, may need to be integrated with BIM platforms. Further, research should investigate how these individual technologies can be integrated into a microservice ecosystem for BIM.

Secondly, while this research demonstrated scalability advantages, the scale of the prototype is small when compared to the potential size of a microservice ecosystem associated with a full construction site. The SOA pattern was chosen for its ability to scale up to tens of thousands of microservices in commercial applications, yet the research here deployed only a small collection. The ecosystem needs to be expanded to a larger size suitable for complex building projects, which may reveal new challenges for research such as performance and computing complexity issues associated with a dense network of services.

Thirdly, a wider range of machine learning applications should be explored to integrate into the microservice ecosystem. One of the advantages of the microservice ecosystem, particularly the ability to develop individual containers, is the capability to develop applications in programming languages independent from other applications. However, one of the limitations identified in developing within the Revit environment was the lack of flexibility of using non-.NET languages. On the contrary, the proposed prototype primarily utilized Python. In order to explore the full flexible capabilities of container technology, it would be beneficial to experiment with different data-processing and machine learning applications operating in a wide variety of languages while working together. Aside from the technical advantages, this will better support my vision of a more diverse and inclusive environment for developers.

Finally, it would be beneficial to collaborate with industry stakeholders. The motivation for this research is derived from industry observations, and while the data is provided by an actual construction site, the software research culminates in a speculative prototype. For implementing and testing the prototype and its application in a professional environment,

collaboration with industry is required. This research has the potential to make several contributions to the field of BIM research and the wider DT research conversation in the AEC industry. By exploring new approaches to BIM data management and sensor integration, the findings of this thesis might pave the way for more efficient and flexible solutions that better accommodate the diverse and often fragmented nature of the industry. Several contributions have been identified as follows:

Bibliography

- [1] Angjeliu, G., Coronelli, D., and Cardani, G. (2020). Development of the simulation model for Digital Twin applications in historical masonry buildings: The integration between numerical and experimental reality. *Computers & Structures*, 238:106282.
- [2] Anurag, G. (2019). Python Implementation of OPTICS (Clustering) Algorithm.
- [3] Azhar, S. (2011). Building Information Modeling (BIM): Trends, Benefits, Risks, and Challenges for the AEC Industry. *Leadership and Management in Engineering*, 11(3):241–252.
- [4] Barbosa, F., Woetzel, J., and Mischke, J. (2017). Reinventing Construction: A Route to Higher Productivity. Technical report, McKinsey Global Institute.
- [5] Boje, C., Guerriero, A., Kubicki, S., and Rezgui, Y. (2020). Towards a semantic Construction Digital Twin: Directions for future research. *Automation in Construction*, 114:103179.
- [6] Bredella, N. (2019). Simulation and Architecture: Mapping Building Information Modeling. *NTM Zeitschrift für Geschichte der Wissenschaften, Technik und Medizin*, 27(4):419–441.
- [7] buildingSMART (2020). Technical Roadmap buildingSMART - Getting ready for the future.
- [8] buildingSMART (2023). IFC Formats.
- [9] Coupry, C., Noblecourt, S., Richard, P., Baudry, D., and Bigaud, D. (2021). BIM-Based Digital Twin and XR Devices to Improve Maintenance Procedures in Smart Buildings: A Literature Review. *Applied Sciences*, 11(15):6810.
- [10] Cureton, P. and Dunn, N. (2021). Digital twins of cities and evasive futures. In *Shaping Smart for Better Cities*, pages 267–282. Elsevier.
- [11] Deng, M., Menassa, C. C., and Kamat, V. R. (2021). From BIM to digital twins: a systematic review of the evolution of intelligent building representations in the AEC-FM industry. *Journal of Information Technology in Construction (ITcon)*, 26(5):58–83.

- [12] Eastman, C. M. (1974). An Outline of the Building Description System. Research Report 50, Carnegie Mellon University.
- [13] Eastman, C. M., Eastman, C., Teicholz, P., Sacks, R., and Liston, K. (2011). *BIM handbook: A guide to building information modeling for owners, managers, designers, engineers and contractors*. John Wiley & Sons, 2011.
- [14] Erdolu, E., Losi, S., and Williams, B. (2022). Digital Twin: A Brief Literature Review and Potential Research Directions for ReAC. Technical report, Carnegie Mellon University.
- [15] Etiido (2020). A Brief History of ArchiCAD.
- [16] Goolsbee, A. and Syverson, C. (2023). The Strange and Awful Path of Productivity in the U.S. Construction Sector. Technical Report 2023-04, University of Chicago Booth School of Business and NBER.
- [17] Gud, A. (2019). Testing in Production at Scale.
- [18] Harper, K. E., Malakuti, S., and Ganz, C. (2019). Digital Twin Architecture and Standards. Medium: 12pp. Publisher: UNESCO/IOC.
- [19] Hou, L., Wu, S., Zhang, G. K., Tan, Y., and Wang, X. (2020). Literature Review of Digital Twins Applications in Construction Workforce Safety. *Applied Sciences*, 11(1):339.
- [20] Hurley, S. (2009). AutoCAD AEC on the Mac (1990) 1.
- [21] Jakobi, M., Kunath, L., Stöckli, U., and Witzig, A. (2018). BIM use-case : model-based performance optimization. Medium: application/pdf Publisher: International Solar Energy Society.
- [22] Kaewunruen, S. and Lian, Q. (2019). Digital twin aided sustainability-based lifecycle management for railway turnout systems. *Journal of Cleaner Production*, 228:1537–1551.
- [23] Khajavi, S. H., Motlagh, N. H., Jaribion, A., Werner, L. C., and Holmström, J. (2019). Digital Twin: Vision, Benefits, Boundaries, and Creation for Buildings. *IEEE Access*, 7:147406–147419.
- [24] Krylovskiy, A., Jahn, M., and Patti, E. (2015). Designing a Smart City Internet of Things Platform with Microservice Architecture. In *2015 3rd International Conference on Future Internet of Things and Cloud*, pages 25–30, Rome, Italy. IEEE.
- [25] Lewis, J. and Fowler, M. (2014). Microservices a definition of this new architectural term.
- [26] Llach, D. C. (2015). *Builders of the Vision: Software and the Imagination of Design*. Routledge, New York, NY, 1st edition edition.

- [27] Logothetis, S., Delinasiou, A., and Stylianidis, E. (2015). Building Information Modelling for Cultural Heritage: A review. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, II-5/W3:177–183.
- [28] Lopes, T. P. and Eracar, Y. A. (2011). TPS development using the Microsoft .NET Framework. In *2011 IEEE AUTOTESTCON*, pages 259–265, Baltimore, MD, USA. IEEE.
- [29] Lu, Q., Xie, X., Parlikad, A. K., Schooling, J. M., and Konstantinou, E. (2021). Moving from building information models to digital twins for operation and maintenance. *Proceedings of the Institution of Civil Engineers - Smart Infrastructure and Construction*, 174(2):46–56.
- [30] M, L. and Kiviniemi, A. (2012). The IFC standard: A review of History, development, and standardization, Information Technology. *ITcon*, 17:134–161.
- [31] Meshenberg, R. (2016). Microservices at Netflix Scale.
- [32] Opoku, D.-G. J., Perera, S., Osei-Kyei, R., and Rashidi, M. (2021a). Digital twin application in the construction industry: A literature review. *Journal of Building Engineering*, 40:102726.
- [33] Opoku, D.-G. J., Perera, S., Osei-Kyei, R., and Rashidi, M. (2021b). Digital twin application in the construction industry: A literature review. *Journal of Building Engineering*, 40:102726.
- [34] Opoku, D.-G. J., Perera, S., Osei-Kyei, R., Rashidi, M., Famakinwa, T., and Bamdad, K. (2022). Drivers for Digital Twin Adoption in the Construction Industry: A Systematic Literature Review. *Buildings*, 12(2):113.
- [35] Ozturk, G. B. (2021). Digital Twin Research in the AECO-FM Industry. *Journal of Building Engineering*, 40:102730.
- [36] Pan, Y. and Zhang, L. (2021). A BIM-data mining integrated digital twin framework for advanced project management. *Automation in Construction*, 124:103564.
- [37] Peng, Y., Zhang, M., Yu, F., Xu, J., and Gao, S. (2020). Digital Twin Hospital Buildings: An Exemplary Case Study through Continuous Lifecycle Integration. *Advances in Civil Engineering*, 2020:1–13.
- [38] Rahman, M. M. (2014). Barriers of Implementing Modern Methods of Construction. *Journal of Management in Engineering*, 30(1):69–77.
- [39] Rainerge, W. (2020). 5 Clustering Algorithms Data Scientists need to know – The key is always to understand the basic approach of any algorithm you want to use.

- [40] Sanjaya, H. (2020). Monolith vs Microservices.
- [41] Saputra, A. (2022). Best Practices for REST API Development.
- [42] Scott Talon Walker Architects, f. (2018). Delivering World-Leading Healthcare Using BIM.
- [43] Tang, S., Shelden, D. R., Eastman, C. M., Pishdad-Bozorgi, P., and Gao, X. (2019). A review of building information modeling (BIM) and the internet of things (IoT) devices integration: Present status and future trends. *Automation in Construction*, 101:127–139.
- [44] Tchana, Y., Ducellier, G., and Remy, S. (2019). Designing a unique Digital Twin for linear infrastructures lifecycle management. *Procedia CIRP*, 84:545–549.
- [45] Vergeest, J. (1996). Product Data Exchange. *Computer-Aided Design*, 28(8):665.
- [46] Villupuram, S. (2020). Bricks, data and mortar: it’s time to build in the intelligence.
- [47] Wang, G. (2011). Improving Data Transmission in Web Applications via the Translation between XML and JSON. In *2011 Third International Conference on Communications and Mobile Computing*, pages 182–185, Qingdao, China. IEEE.
- [48] Wang, W., Hu, H., Zhang, J. C., and Hu, Z. (2020). Digital Twin-based Framework for Green Building Maintenance System. In *2020 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, pages 1301–1305.
- [49] Weichenberger, F. P., Schwaiger, C., and Höfer-Öllinger, G. (2020). Von der geologischen Aufnahme zur BIM-Repräsentation. *Geomechanics and Tunnelling*, 13(2):199–211.
- [50] Woodhead, R., Stephenson, P., and Morrey, D. (2018). Digital construction: From point solutions to IoT ecosystem. *Automation in Construction*, 93:35–46.
- [51] Yeong, L. (2019). DBSCAN Density-Based-Spatial-Clustering-of-Applications-with-Noise.
- [52] Zhao, X. (2017). A scientometric review of global BIM research: Analysis and visualization. *Automation in Construction*, 80:37–47.