Challenges in Converting Building Information Models into Virtual Environments for FM Operations and User Studies in the Built Environment

Sutenee Nopachinda[1] and Semiha Ergan[2]

1) Undergraduate Student, Department of Civil and Urban Engineering, NYU Tandon School of Engineering, Six MetroTech Center, Brooklyn, NY, USA. Email: sutenee@nyu.edu
2) Prof., Department of Civil and Urban Engineering, NYU Tandon School of Engineering, Six MetroTech Center, Brooklyn, NY, USA. Email: semiha@nyu.edu

Abstract:
Facilities management (FM) in large scale buildings requires ample coordination of space, assets, and maintenance information to ensure that the needs of the building are met in the most cost efficient way possible. However, it is difficult to keep track of updates in facilities, causing an increase in facility-related expenditures due to idle waiting for information verification. Virtual environments, digital representation of physical spaces in a virtual model, provide opportunities to access to facility information in an integrated, visual, and 1-1 scale environment. To create these environments, building information models (BIMs) are converted into virtual models through BIM tools, game engines, and virtual reality software. However, there are numerous challenges with the current practice. This paper provides an analysis of alternative workflows to generate virtual models from BIMs and the challenges faced during the conversion process. The analysis is performed using commercially available BIM authoring tools, gaming engines, and tools specialized for immersive virtual representations. The case building is an existing eleven floor academic building that was undergoing major renovations. This paper also discusses the issues and challenges faced while generating virtual models for FM applications and user studies in the built environment. The research showed that various issues exist in the workflow such as data loss in components, time loss due to non-valuing tasks in the workflow, model transformations, and the lack of reasoning mechanisms that enable interactions with components. Based on these setbacks, suggestions are made to improve the existing workflows and minimize the identified issues.

Keywords: Building information modeling, BIM, virtual environments, game engines, facility management

1. INTRODUCTION

Wider adoption of building information models (BIMs) within the Architectural/Engineering/Construction (AEC) industry has resulted in wider recognition and use of BIMs for facilities management (FM). Integration between BIMs (representing static facility information) and building automation systems (representing dynamic facility information) has been increasingly discussed and under development. One of the major challenges faced with using BIM during the FM phase is the integration of model use for daily work processes. Virtual worlds, digital representation of physical spaces in a virtual model, provide opportunities to use BIMs and access to facility information in an integrated, navigable, and 1-1 scale environment for both FM operations and user studies in the built environment. Such virtual worlds are possible by converting existing BIMs into gaming engines or in virtual reality (VR).

Gaming engines are frameworks which allow the development of video games, virtual worlds, and interactive 3D environments. For FM and BIM use, they are used to provide facility information on various types of components such as walls, and systems including mechanical and structural members. To simulate navigation and object interaction in virtual worlds, game engines provide tool sets for first-person navigation and script development to display building system information when interacted. The virtual world can then be exported to a virtual reality application for navigating the building with motion-capture devices. While these tools would greatly improve the workflow for facility managers and users studies in the built environment, there are several challenges in converting BIMs to virtual worlds. Within the current practice, virtual worlds are mainly studied in software and game development fields. Since these fields rely on models developed directly within utilized gaming environments, the challenges in converting BIMs to virtual worlds have yet to be studied.

This paper provides an analysis of alternative workflows to generate virtual worlds from BIMs and the challenges faced when following these processes. Analysis is performed by using commercially available BIM authoring tools, gaming engines, and tools specialized for immersive virtual representations. The BIM used in this case study is an existing eleven floor academic building that was undergoing major renovations. The following sections discuss related research on the current challenges in creating virtual environments for AEC/FM applications, the methodology of this study, and our findings. Suggestions are made to help streamline the process of converting BIMs into virtual environments and the standardizations necessary to create an efficient workflow for industry use.

2. BACKGROUND RESEARCH

Immersive virtual models and environments have been used in the AEC industry for various areas. Applications include construction planning (Messner, 2006; Savioja et al., 2003), energy retrofitting for buildings (Yang, et al., 2013), and general involvement between project players (Leicht et al., 2010). These studies have highlighted advantages of using virtual models in these application areas as compared to the current practice. For the same advantages, virtual models have been studied for FM applications such as resolving work orders, determining problematic equipment, or undergoing general systems maintenance. Studies have shown that displaying building automation systems data in a more intuitive format significantly decreases the time it takes to resolve issues by 26 to 65% compared to the existing interfaces (Yang & Ergan, 2014; Yang & Ergan, 2015).

BIMs can be incorporated with immersive elements through gaming engines such as CryENGINE, Unity3D, and Unreal Engine. Numerous studies have assessed several workflows for converting BIMs into virtual environments using different gaming engines or VR systems. Currently, the general workflow for this process is to take an existing BIM and import it into a gaming engine or VR system (Dalton & Parfitt, 2013; Billie et al., 2014). However, depending on the BIM tool used and the end use, several intermediate steps are required such as using plug-ins to convert the BIM directly into the VR system (Johansson et al., 2014) or rendering the BIM in a rendering software before importing it into a game engine (Dalton & Parfitt, 2013).

Literature highlights the main issues encountered in the model conversion process, such as the lack of interoperability between software (Billie et al., 2014) and software unresponsiveness when using large models in the workflow (Dalton & Parfitt, 2013). Lack of interoperability contributes to loss of textures and meta-data for components when importing BIMs directly into a game engine (Billie et al., 2014). With large models, most of the model's components are typically created in complex geometry which makes the model more data-heavy (Billie et al., 2014). This causes software to lag, become unresponsive, or miss textures during rendering (Billie et al., 2014). Several studies have tried to bypass the conversions between software to prevent data loss, such as enabling a CAVE system directly from a BIM tool (Kang, Ganapathi and Nseir, 2012), using rendering plug-ins in the BIM tool, and directly exporting into a VR tool (Johansson et al., 2014).

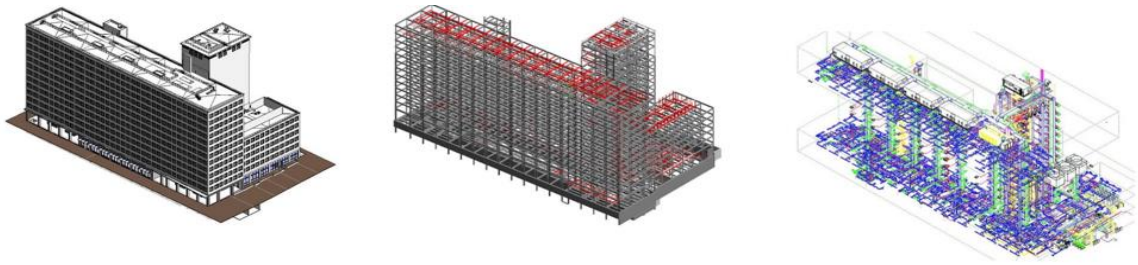
Previous research studies provide a representative set of examples for applications of virtual environments to AEC/FM work tasks. However, they fail to provide a point of departure for defining the required intensity of geometric representation in BIMs for FM applications and achieving responsive conversions. This study will report the challenges faced when using a data-heavy BIM, identify challenges in generating virtual worlds from BIMs for FM applications, and suggest methods to create a more efficient workflow during the conversion process.

3. METHOD

The objective of this research is to conduct experiments to determine the intensity of geometric detail needed in BIMs for a virtual model for FM use, identify challenges in generating virtual worlds from BIMs for FM applications, and recommendations to define a more efficient workflow. Several experiments were conducted using an academic building's BIM and facilities management data including BAS dynamic data and maintenance schedules. Experiments include examining the effect of model sizes and it's granularity on time efficiency for the BIM to virtual environment process, determining efficient methods to integrate HVAC data from outside sources into the environment, and creating spatial relationships with each component in the model.

The case study model used is a thirteen story academic building currently undergoing renovation. Models of several engineering disciplines including architectural, mechanical, and fire protection were provided in a popular BIM modeling software format. Each were merged into one central model to be converted into a virtual world. Since the model was bulky, one of the floors that were detailed in design for renovation was used for the experiments. The 11th floor was isolated by removing components from every floor except the major HVAC equipment and their connections in other floors that fed the 11^{th} floor. Figure 1 shows the types of components contained in the model according to their discipline.

Additional data sources include the building automation systems (BAS) data and interface for HVAC equipment in various buildings owned by the same owner of the case building. Data includes live utility consumption for all equipment such as heating and cooling air temperatures in air conditioning units, current and set point air temperatures for every room, and system performance measures. These data items were integrated in the virtual environment and displayed in a visual representation for FM use. Since the building was undergoing renovation, no BAS data was available. Hence the research team used another building's BAS data to use in the experiments for identifying challenges faced in integrating facility dynamic data in virtual worlds along with BIMs. Several platforms were used in the experiments to identify alternative work processes for converting BIMs to virtual worlds including BIM/project modeling/viewing software, rendering software for BIMs, gaming engines, and VR software.



(a) Architectural (b) Structural (c) Mechanical

Figure 1. Snapshots from BIM of the case building for selected building systems

4. ALTERNATIVE WORKFLOWS FOR CONVERTING BIMS TO VIRTUAL WORLDS

We have identified three different workflows for converting BIMs to virtual worlds. Figure 2 shows Workflow #1 which consists of importing a BIM into rendering software before exporting into a game engine and VR software. This workflow is popular due to the fact that many BIMs are created with the software listed in Figure 2. Figure 3 shows Workflow #2, which directly imports a BIM into VR software through commercial plug-ins. The commercial plug-ins in this process are not native to the BIM software. Plug-ins typically take the views within the BIM software and convert them into 3D models, allowing navigation in either a first person or third person perspective. Figure 4 shows Workflow #3, which bypasses the rendering step in Workflow #1 and the BIM is directly imported into the game engine, and then in the VR software.

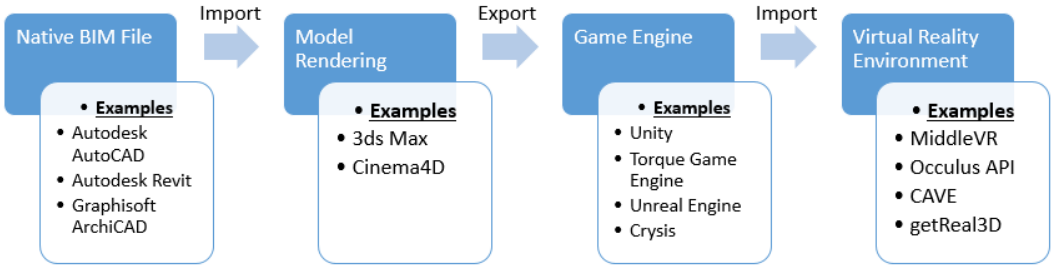


Figure 2. Workflow #1: Converting rendered BIM to gaming engine and VR environments

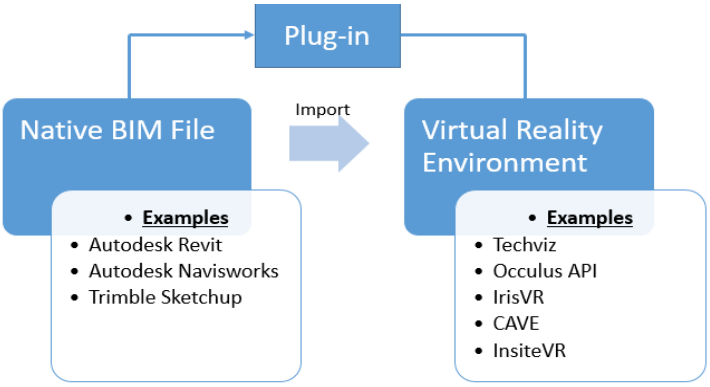


Figure 3. Workflow #2: Directly importing BIM into VR environment via plug-in

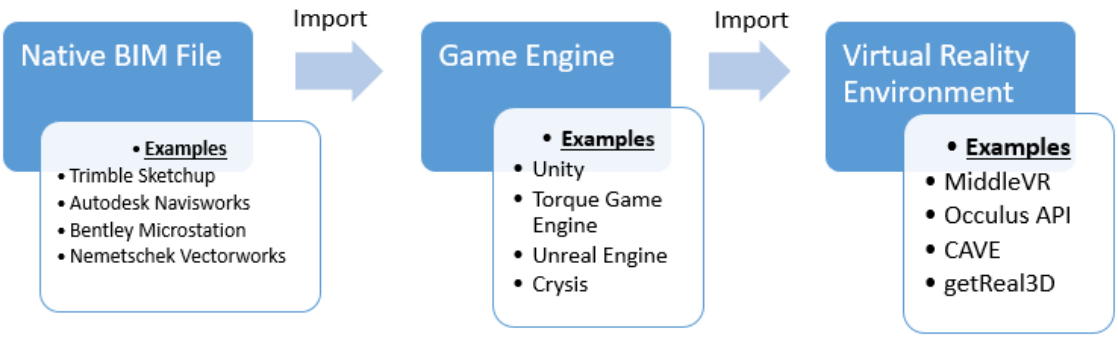


Figure 4. Workflow #3: Importing BIM into gaming engine and VR environment

4.1 Importing Native BIM Files to Rendering Software

This first step is required for Workflow #1 as some gaming engines do not recognize the model's textures when BIM files are immediately imported into it. The native BIM files are exported in a file format that is recognized by model rendering software (e.g., 3ds Max). When the necessary lighting and texture information is rendered, it is exported into the game engine.

4.2 Adding Model Interactivity in Game Engines

The rendered models are then imported into a game engine, as shown in **Error! Reference source not found.**5. When imported, the option, "Enable Colliders", gave all components a script that defines its physical collisions. Any textures that are not rendered in the rendering environment can be re-applied in the game engine environment. A first-person character controller is then imported to enable navigation within the building. In the experiments, scripts for the game GUI were written in Javascript. The GUI consists of a menu to switch between different locations in the model called "scenes", pop-up windows that display a component's information when clicked on, a distance indicator that points to the nearest components of interest, and a component highlighter that highlights the components of interest around the player (see Figure *5*).

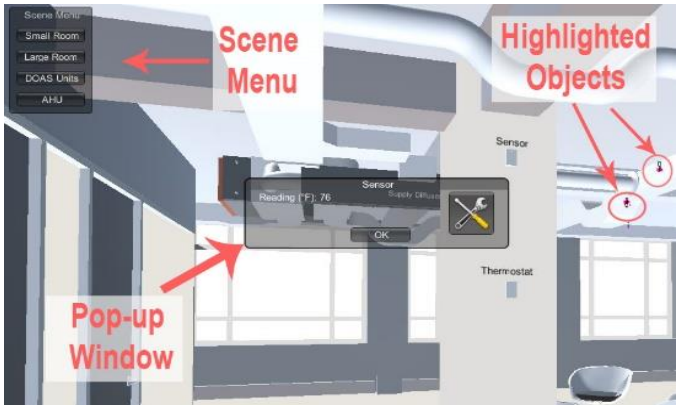


Figure 5. A model snapshot in the utilized game engine

There are different methods to bring data from outside sources into the game engine. For this study, a SQL database was created to contain BAS data for numerous HVAC components including air conditioning units and cooling towers. Scripts written in C#, Javascript, and Python were used to automatically process comma separated value (CSV) files into the database and to pull data for respective components into the game engine dynamically so that displayed data in the game engine was updated as new data streamed into the database. Figure *6* summarizes this process.

Figure 6. Process to integrate databases in the game engine

Table 1 shows the scripts created to enable this virtual model and their functions. Other methods of including data are interfacing with BAS API or servers that directly pull BAS data for FM storage purposes.

Table 1. Scripts created for generating the virtual world for FM use

Script Function	Language	Description
BAS data to SQL Database	Python	Parses CSV files of BAS data for individual components into SQL database.
Connect SQL database to Unity model	C#	Provides a connection between SQL database and the game engine to extract data for individual components
Display component information	C#	Extracts data specific to the component from the database. Provides interactivity for components to allow users to display data on screen.
Display user interface	Javascript	Shows the user options to switch between locations of interest within the model. Highlights objects of interest and shows their relative distance to the user.

4.3 Converting Game Model into VR Environment

To enable a virtual reality environment for the model from the gaming engine a VR software was used. The VR package is imported into the game engine. The VR manager component is added in the project Hierarchy and the game is built as a Windows application. The application is run through the VR configurator which enables the VR environment. The model in the VR setting is shown in Figure *7*. Using the workflow described in this section, the authors conducted experiments with the case model and findings are provided in the next section.

Figure 7. Model in VR Environment

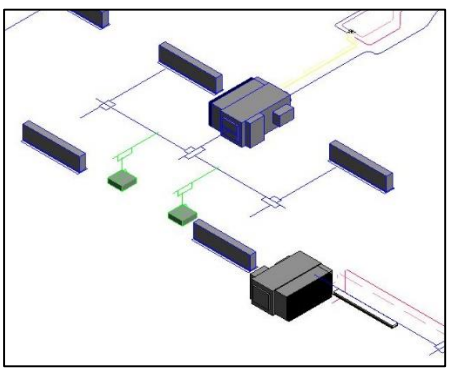
5. RESEARCH FINDINGS AND LESSONS LEARNED FROM THE EXPERIMENTS

5.1 Model Granularity Effects Conversion Time and Model Quality in Virtual Environments

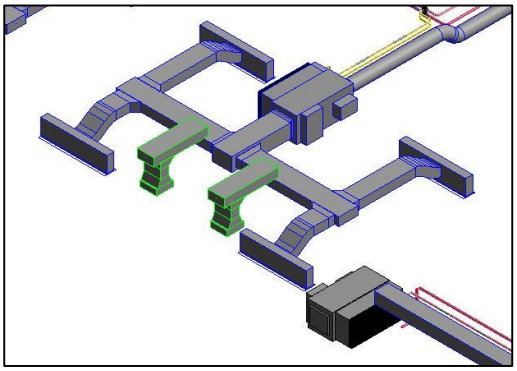
When importing the BIMs into the rendering environment, it was exported at three levels of geometric granularity (LGG): Coarse, Medium, and Fine. Each level of geometric granularity signifies the intensity of detail for the model's components with Coarse as the lowest detail and Fine as the highest detail. It also determines the amount of objects to reduce in order to reduce rendering time. By default, components modeled in a BIM authoring tool are Medium LGG and can be converted into Coarse or Fine. These levels directly alter the size of the model, as each component's geometry contains simpler shapes and less vertices than models with Fine detail. However, the heavier the model is, the slower the runtime will be for rendering in the rendering, game engine, and VR steps. Therefore, understanding the effect of a model's LGG is important to optimize the model's details in the VR environment and reduce the amount of lag that would be faced during the rendering steps.

An analysis was made to determine the effect of different LGG and its effect on file size, polygon count, and the number of vertices of the model's components. Polygon counts and vertices determine how a component is rendered and the amount of detail it shows. Our analysis showed that LGG the Coarse level deleted a significant amount of components or rendered several components with the barest amount of vertices and polygons, as seen in Figure 8(a).

Rendering the BIM with an LGG of Fine greatly increased the amount of polygon counts and vertices for each components and the file size when compared to LGGs of Fine and Medium. Figure 8(b) and Figure *9* show the BIM in coarse and fine LGG in the modeling tool and after rendering in the game engine, respectively. An LGG of Medium bared similar resemblance to a LGG of fine and is not shown. When rendered with a Coarse LGG, components' physical properties were lost and rendered as a single line. The model with a fine LGG was rendered properly with components staying intact.

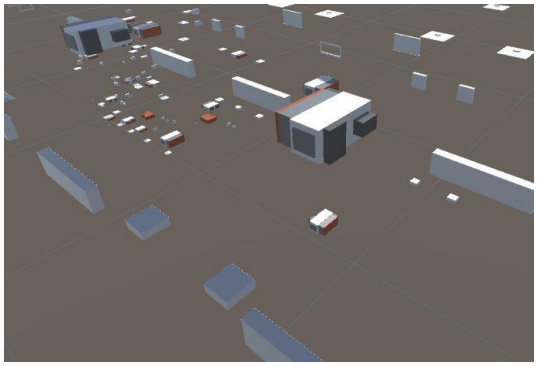


(a) Coarse LGG

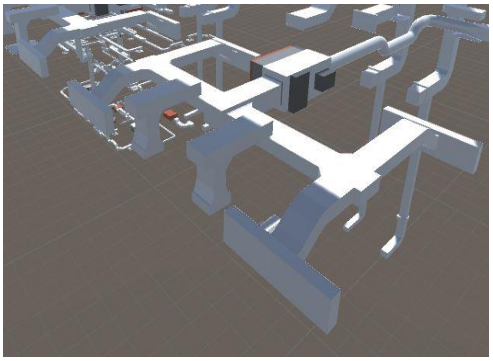


(b) Medium/Fine LGG

Figure 8. LGGs in the modeling tool



(a) Coarse LGG



(b) Medium/Fine LG

Figure 9. LGGs in the game engine after rendering

Table 22 shows the file size for each LGG after the model is exported from the BIM authoring tool and after the model was rendered in the rendering software. Based on this data, as the LGG of the model increases, obviously file size increases as well. Referring to Table 2, file size increases by a factor of two from Coarse LGG to Medium LGG and by a factor of four from Medium LGG to Fine LGG. File size dramatically increases after rendering textures and lighting by at least two times the original size for each LGG.

Table 2. File sizes for different LGGs after applied renderings

Software	**File Size (KB)**		
	Coarse	Medium	Fine
BIM authoring tool	99,376	169,268	636,699
Rendering tool	269,800	455,896	1,313,169

The number of polygons and vertices also increases as LGG increases as seen in Table 3. The number of polygons and vertices dramatically increases by a factor of 3.5 between Medium and Fine LGGs. The numbers in Tables 2

and 3 are specific to the utilized case model and depends on the model size. However, the ratios between the levels of geometric details hold valid.

Table 3. Number of polygons and vertices for different LGGs

LGG	Polygon Count	Number of Vertices
Coarse	9,835,028	5,033,914
Medium	12,059,488	6,357,340
Fine	31,971,517	17,091,049

Table 4 shows the rendering time from the default Medium LGG to Coarse or Fine LGG in the BIM tool, rendering tool, gaming engine and VR tool. There is no rendering time recorded for Medium in Table for BIM and rendering tools, as Medium was the default LGG when the model was created. Modifying the LGG in the BIM tool was much easier and faster than in the rendering tool, as the BIM was not rendered and was more lightweight in terms of data. Table 4 also shows the time to import the model and render it within the game engine. It is evident that the heavier the model is, the more time it takes to import and render the model in the game engine. Software unresponsiveness and lag was faced when importing the BIM with Fine LGG into the game engine. Table 4 also shows the rendering time for VR tools. Based on these data, it is clear that the larger or data heavy the BIM is, the more time it takes to render and import the BIM in either the game engine or VR tool. Additionally, heavier models will cause a large amount of lag and software unresponsiveness during the workflow. This is a major drawback for the workflow, since typically large models would be used for FM use.

Table 4. Rendering time in modeling, rendering, game engine and VR tools for different LGGs

Time in Specific tools	Level of Geometric Granularity		
	Coarse	Medium	Fine
Rendering Time In BIM Tool	8 sec	-	14 sec
Rendering Time In Rendering Tool	3 min	-	13 min
Rendering Time In Game Engine	30 sec	1 min 30 sec	7 min
Importing Time in Game Engine	12 min	16 min	55 min 30 sec
Rendering Time in VR tool	3 min	8 min	15 min

Based on the data in Tables 2 and 3, a model with a medium LGG is the most optimal due to the file size and polygon difference as compared to a fine LGG. From the analysis, choosing an appropriate LGG for a BIM will help create a more time efficient workflow from converting a BIM into a VR environment and portray the right amount of details when the environment is used by facility managers. A model rendered with the most optimal LGG would help with future steps in the workflow, such as rendering and enabling data feeds into the environment. Choosing an LGG which renders objects with minimal detail will not accurately show the components to facility managers and can potentially inhibit their decision making process from the inability to display data visually.

5.2 Loss of Semantic Information in BIMs when Converted To Virtual Worlds

One of the most prominent challenges faced was data loss in components throughout the workflow and identifying potential solutions to import lost data into gaming engines. The first type of data lost are textures. For the BIM tool used in this study, the gaming engine used did not recognize encrypted material libraries in the imported file from the BIM tool. Component specific color and textures are lost when directly importing the BIM into the gaming engine without rendering. Enabling colors and textures requires the intermediate step of rendering the BIM in rendering software before importing it into a gaming engine. This step is time consuming and increases the file size of the model, especially for large and complex models. Additionally, because of the slow response time during rendering, not all textures were rendered when the model was opened in the gaming engine.

The second type of data loss is semantic information about building components. Semantic information are specific characteristics of the components including equipment purpose, weight, and its specific attributes. For example, a water pump object in the BIM would have attributes such as connection size, flow rate, and rotation speed. When the BIM is exported into the rendering software or the game engine, the semantic data for each component is lost. This is a problem when defining spatial relationships between components, such as identifying which valves are in certain rooms or how shutting off a specific air conditioner will affect the area around it. One possible solution is to integrate Industry Foundation Classes (IFC) data for each component into the gaming engine using a workflow similar to integrating BAS data into the game model and currently being investigated for feasibility.

5.3 Lack of mechanism to Link BAS Data in Virtual Worlds

While establishing a connection between BAS Data and the game engine was feasible, creating an automatic live

data feed proved problematic. There are multiple ways to achieve this functionality including interfacing with BAS application program interface (API) to extract server data or pulling data from an established server connected to BAS. A bottleneck was faced when linking components in the game engine to its respective data. Since there is no map between equipment IDs and parameters in automation system and BIM component IDs, it is not possible to create an automatic link between data and components. Since the current naming conventions in the modeling environment are different from the systems controls environment, it is not possible to relate BAS data directly to the model. Therefore, each item in the database needs be manually linked one-to-one to the component via scripts.

5. CONCLUSIONS

This paper details the challenges faced when converting a BIM into a virtual environment for FM purposes. The process followed includes importing a case study BIM into a rendering tool for color, texture, and lighting rendering, applying scripts on the model's components in a gaming engine to create an interactive model, and importing into a VR tool to establish a VR environment. During the workflow, the most prominent issues were software unresponsiveness due to large file size, difficulty in establishing an automatic link between BAS control systems and our virtual environment, and data loss in components. Large models suffer from software unresponsiveness due to a massive amount of polygons and components to render. Three different level of geometric granularity were explored for the model and their outputs were analyzed.

Issues were also faced when establishing a data link between BAS control systems and the model, such as the inability to interface with the BAS system to automatically pull data into a database and create an automatic link between data and component. In future research, implementing IFC elements into gaming engines and the feasibility of establishing relationships between BIM and BAS through IFC in virtual environments and linking semantic component information to virtual components in VR environments will be explored.

REFERENCES

Bille, R., Smith, S., Maund, S., and Brewer, G. (2014). Exploring building information modeling (BIM) to game engine conversion, *Working Paper Series*, *Number 7*. http://nova.newcastle.edu.au/vital/access/manager/Repository/uon:14123

Dalton, B and Parfitt, M. (2013). Immersive visualization of building information models, *Design Innovation Research Centre working paper 6*, Version 1.0, https://www.reading.ac.uk/web/FILES/designinnovation/DIRC_Working_Paper_6.pdf

Johansson, M., Roupé, M., and Tallgren, M. (2014) From BIM to VR - Integrating immersive visualizations in the current design process, *Thompson, Emine Mine (ed.), Fusion - Proceedings of the 32nd eCAADe Conference - Volume 2, Department of Architecture and Built Environment, Faculty of Engineering and Environment*, September 10-12, 2014, Newcastle upon Tyne, England, UK, pp. 261-269.

Kang, J., Ganapathi, A., Nseir, H. (2012) Computer aided immersive virtual environment for BIM, *14th International Conference on Computing in Civil and Building Engineering*, June 27-29, 2012, Moscow, Russia.

Leicht, R. M., Kumar, S., Abdelkarim, and Messner, J. (2010). Gaining End User Involvement through Virtual Reality Mock-Ups: A Medical Facility Case Study, *CIB W78 2010 - Applications of IT in the AEC Industry*, http://itc.scix.net/cgi-bin/works/Show?w78-2010-143

Messner, J. (2006) Evaluating the Use of Immersive Display Media for Construction Planning, *In Proceedings of 13th EG-ICE Workshop, Intelligent Computing in Engineering and Architecture*, June 25–30 2006, Ascona, Switzerland, pp. 484-491. http://ezproxy.library.nyu.edu:2178/chapter/10.1007/11888598_43

Savioja, L., Mantere, M., Olli, I., Äyräväinen, S., Gröhn, M., and Iso-aho, J. (2003) Utilizing virtual environments in construction projects, *ITcon Vol. 8, Special Issue Virtual Reality Technology in Architecture and Construction*, pp. 85-99, http://www.itcon.org/2003/7

Yang, X. and Ergan, S. (2014). Evaluation of visualization techniques for use by facility operators during monitoring tasks, *Automation in Construction*, 44 (August), pp. 103–118. DOI: 10.1016/j.autcon.2014.03.023.

Yang, X. and Ergan, S. (2015). Design and evaluation of an integrated visualization platform to support corrective maintenance of HVAC problem-related work orders, J*ournal of Computing in Civil Engineering*, DOI: 10.1061/(ASCE)CP.1943-5487.0000510 , 04015041.

Yang, X., Liu, Y., Ergan, S., Akinci, B., Leicht, R., and Messner, J. (2013) Lessons Learned from Developing Immersive Virtual Mock-Ups to Support Energy-Efficient Retrofit Decision Making, *Computing in Civil Engineering (2013)*, pp. 210-217, DOI: 10.1061/9780784413029.027.