LOW CARBON LIVING CRC

# Progressing precinct modelling on the UNSW campus and beyond: BIM/PIM and 3DGIS

## System architecture and Visualisation

| | |
|---|---|
| Authors | Mitko Aleksandrov, Abdoulaye Diakité, Jinjin Yan, Wei Li, Jack Barton, Sisi Zlatanova |
| Title | Progressing precinct modelling on the UNSW campus and beyond: BIM/PIM and 3DGIS: System Architecture and Visualisation |
| ISBN | |
| Date | May 2019 |
| Keywords | 3D, PIM, geospatial data, BIM, database, visualisation |
| Publisher | CRC for Low Carbon Living |
| Preferred citation | Aleksandrov, M., A. Diakité., J. Yan, W. Li., S. Zlatanova, 2019 Progressing precinct modelling on the UNSW campus and beyond: BIM/PIM and 3DGIS System Architecture and Visualisation, CRC Low Carbon Living report, 27p. |

Australian Government
Department of Industry, Innovation and Science

Business Cooperative Research Centres Programme

LOW CARBON LIVING CRC

## Acknowledgements

This research is funded by the CRC for Low Carbon Living Ltd supported by the Cooperative Research Centres program, an Australian Government initiative

## Disclaimer

Any opinions expressed in this document are those of the authors. They do not purport to reflect the opinions or views of the CRCLCL or its partners, agents or employees.

The CRCLCL gives no warranty or assurance and makes no representation as to the accuracy or reliability of any information or advice contained in this document, or that it is suitable for any intended use. The CRCLCL, its partners, agents and employees, disclaim any and all liability for any errors or omissions or in respect of anything or the consequences of anything done or omitted to be done in reliance upon the whole or any part of this document.

## Peer Review Statement

The CRCLCL recognises the value of knowledge exchange and the importance of objective peer review. It is committed to encouraging and supporting its research teams in this regard.

The authors confirm that this document has been reviewed and approved by the project's steering committee and by its program leader. These reviewers evaluated its:

- originality
- methodology
- rigour
- compliance with ethical guidelines
- conclusions against results
- conformity with the principles of the Australian Code for the Responsible Conduct of Research (NHMRC 2007),

and provided constructive feedback which was considered and addressed by the author(s).

# Contents

List of Tables ..... 4
List of Figures ..... 5
Acronyms ..... 6
Executive Summary ..... 7
Introduction ..... 8
System architecture ..... 9
Data pre-processing ..... 10
3D object reconstruction ..... 10
BIM pre-processing ..... 11
From IFC to PostGIS ..... 12
Georeferencing the BIM models ..... 12
Preparing the 2D data ..... 13
Data structuring ..... 15
3D Modelling ..... 15
Relational Database Schema ..... 16
Data visualisation ..... 19
Web servers and services ..... 19
3D Visualisation using Cesium ..... 19
Discussion ..... 23
References ..... 24

# List of Tables

Table 1 Data type mapping ..... 16
Table 2 Building table ..... 16
Table 3 Terrain table ..... 17
Table 4 Greensense table ..... 17
Table 5 Archibus table ..... 17
Table 6 Tree table ..... 17
Table 7 Lawn table ..... 17
Table 8 Bim table ..... 17
Table 9 Road table ..... 17
Table 10 Myair table ..... 17

## List of Figures


Figure 1 Map of EM precincts ..... **Error! Bookmark not defined.**

Figure 2 Red Centre electricity usage (a) and water consumption (b)..... **Error! Bookmark not defined.**

Figure 3 List of floorplans ..... **Error! Bookmark not defined.**

Figure 4 Examples of electrical and mechanical data: 2D chilled water diagram- library stage 2 (up), main library ice storage in 3D (down)..... **Error! Bookmark not defined.**

Figure 5 UNSW cadastre parcel..... **Error! Bookmark not defined.**

Figure 6 Footprint (in brown) and above footprint (in white) of buildings..... **Error! Bookmark not defined.**

Figure 7 UNSW walls and fences..... **Error! Bookmark not defined.**

Figure 8 Gardens (a) and vegetation (b) ..... **Error! Bookmark not defined.**

Figure 9 UNSW Kensington sewer layers in AutoCAD..... **Error! Bookmark not defined.**

Figure 10 BIM model in a BIM viewer extracted from ORACLE ..... **Error! Bookmark not defined.**

Figure 11 Red Centre Building, 4th floor ..... **Error! Bookmark not defined.**

Figure 12 Red Centre BIM model front view (up) and UNSW Library BIM model (down) .. **Error! Bookmark not defined.**

Figure 13 Electrical Engineering BIM model (up) and Block house BIM model (down)..... **Error! Bookmark not defined.**

Figure 14 Chemical Science Engineering BIM model (up) and Dalton Building BIM model (down) ... **Error! Bookmark not defined.**

Figure 15 Round house BIM model (up) and a furniture (cupboard) in Roundhouse BIM model (down)**Error! Bookmark not defined.**

Figure 16 Illustration of element furniture in the models in three different buildings..... **Error! Bookmark not defined.**

Figure 17 Electricity use for 2016..... **Error! Bookmark not defined.**

Figure 18 UNSW buildings included in Greensense ..... **Error! Bookmark not defined.**

Figure 19 Top view of UNSW airborne Lidar DSM..... **Error! Bookmark not defined.**

Figure 20 Profile view of the airborne Lidar over UNSW ..... **Error! Bookmark not defined.**

Figure 21 Two snapshots of the point clouds around the Red Centre Building..... **Error! Bookmark not defined.**

Figure 22 Representation of air quality data visualisation in Myair..... **Error! Bookmark not defined.**

Figure 23 Conceptual 3D data model (extended from Beetz et al 2014)..... **Error! Bookmark not defined.**

Figure 24 Generic System Architecture..... **Error! Bookmark not defined.**

## Acronyms

| | |
|---|---|
| BIM | Building Information Modelling |
| CAD | Computer Aided Design |
| DBMS | Database Management System |
| DTM | Digital Terrain Model |
| GML | Geographical Modelling Language |
| IFC | Industry Foundation Classes |
| json | JavaScript Object Notation |
| LoD | Level of Detail |
| OGC | Open Geospatial Consortium |
| OSM | Open Street Map |
| SQL | Structured Query Language |
| UML | Unified Modelling Language |

## Executive Summary

This report presents a way of structuring and visualising geospatial data and their corresponding semantics and metadata in a Precinct Information Model (PIM). The datasets obtained from UNSW Estate Management (EM) are used in this development. The study is completed through several steps involving pre-processing and storage of data, along with web-based visualisation.

## Introduction

The UNSW Kensington Campus lies at the heart of an extended precinct which is the subject of intensive strategic planning and assessment as part of both the Greater Sydney Commission Randwick Collaboration Area and an emerging living laboratory under Randwick City Council's Smart Cities Strategy. The management of the University is keen to develop an example of a 'smart campus' with intelligent structuring and management of geospatial data by employing open standards. Therefore, the campus was selected as a use case for developing a Precinct Information Model (PIM) and corresponding tools for management and visualisation. The data provided by Estate Management of UNSW.

UNSW EM uses many spatial and non-spatial datasets in their daily work. However, the integration and maintenance of these data is very problematic. Data and models are scattered in different file formats and layers, and maintained in a variety of software packages by different departments and institutions. This complicates the update of data and the use and re-use of information.

This report is the first step to understand the options for 2D and 3D geospatial data management for administrative structure of UNSW EM. The study was completed through several steps involving pre-processing and storage of data, along with web visualisation.

The main goal of this report is to explain the process of constructing 3D datasets and propose a unified data structure connecting properties and relationships of different geospatial data in a relational database. Data containing different semantics (i.e., building, stair, grass area, tree, room, wall, roof), geometric types (point, line, polygon, solid, point cloud) and properties were structured according to the international standards CityGML (CityGML, 2019) and IFC (IFC, 2019). For the purpose many pre-processing steps were needed. An open web-based platform Cesium was used for visualisation of the 3D data (Cesium, 2019). In this way, a complete data management system is proposed to model all required steps for successful long-term geospatial data management of PIM, in our case UNSW EM.

### *Available data*

As discussed in the previous report (Zlatanova et al, 2019) several geospatial datasets were obtained from UNSW EM and other publicly available sources. 2D datasets considered in this report are trees, pedestrian navigation areas and green space. Available 3D datasets were only BIM models of several buildings. All other 3D datasets were created within this project. These include extruded representation (LOD1) of buildings showing all buildings at the UNSW campus. Other available datasets considered were point clouds (i.e., unstructured representation of data represented as 3D points), energy data showing energy consumption for several buildings considering period of 3 months and air quality data of some rooms within Red Centre building coming from MyAir system. For more information of available data check the inventory report.

### *Data pre-processing*

Data pre-processing was one of the main parts to establish the right connections between different data structures and their properties. For this purpose, various steps were taken to represent data correctly in a database and be able to visualise them afterwards in a viewer. The pre-processing involved 3D object reconstruction, terrain modelling as well as BIM georeferencing and import. It should be mention that all 2D datasets were transferred to 3D data to model their physical representation realistically on the terrain.

### *Data structuring in a RDBMS*

Since we are dealing with different data structures containing different properties and characteristics, a dedicated data structure was developed. As an object-rational database management system PostgreSQL was selected (PostgreSQL, 2019). The main reason for choosing it is the comprehensive support of various extensions supporting import, manipulation and structuring different geospatial data types such as points, lines, polygons, solids and point clouds. At the same time, it represents an open-source relational database allowing modelling of required relationships between objects. As a result, a unified model is proposed considering available data.

### *Data visualisation*

The next step was visualisation of previously structured data. As a suitable platform which supports visualisation of different geospatial data types Cesium was selected. Cesium is a web-based open-source platform supporting visualisation of complex 2D and 3D geospatial geometries, along with their properties. To establish the connection between Postgres and Cesium, GeoServer was used for streaming simple geometries from the database such as trees and terrain (Geoserver, 2019). For BIM models, extruded buildings and other data, the py3dtiles library was used for the exporting data into 3Dtiles (Py3dtile, 2019), which is the native data format for streaming complex geometries.

The report is organised as follows: Section 2 outlines the proposed system architecture; Section 3 presents the process of data pre-processing; Section 4 discusses data structuring in a database; Section 5 explains the steps taken to visualise the spatial and non-spatial data; Section 6 concludes the study and provides potential improvements for the system.

## System architecture

In order to put in place a system that manages geospatial data in a consistent and coherent way, an appropriate system architecture is needed. Several component layers should be identified including available datasets and datatypes, a method of storing and structuring data, acquisition and exchange of structured data as well as data visualisation and manipulation. Four components were determined (Fig 1).

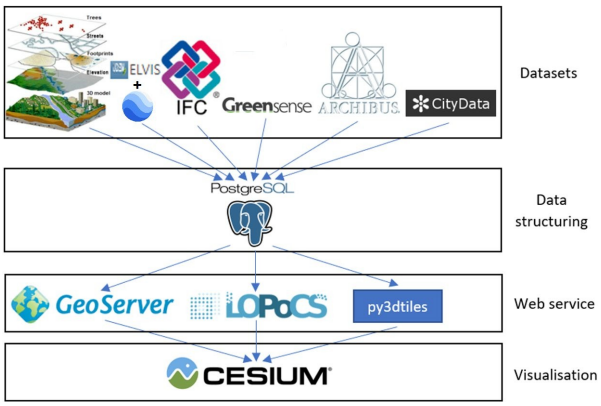



Figure 1 System architecture

The first one tries to understand available geospatial datasets and their geometric types (e.g., point, line, polygon, solid, point cloud), along with their attributes which would enable connection between different datasets and highlight their properties. We classified the data into six groups including 3D datasets, point clouds, BIM models, Greensense (i.e., energy consumption per building) (Greensense, 2019), Archibus data (e.g., metadata of rooms) (Archibus, 2019) and CityData (i.e., air quality data) (CityData, 2019). Since one part of the UNSW campus is on a hill compared to the other which is mainly flat, the available datasets which were predominantly in 2D were transferred into 3D for realistic representation of the space. Point clouds of the campus were obtained from ELVIS online system representing a LiDAR scanning (ELVIS, 2017). For the purpose of visualisation, the points were coloured using the Google Earth images for the same area. All obtained BIM models were in local coordinate system and therefore georeferencing was required to place them in correct position geographically. The other datasets did not require any pre-processing.

The second step represents the selection of a database that enables a native storage of different spatial data types with minimal additional pre-processing. In this regard, PostgreSQL database was selected providing an option to store all identified geometric data types. As mentioned previously, it represents an open-source relational database, having constant upgrades and bringing new extensions which facilitate work with spatial and non-spatial data as well as giving an option for personal upgrades if needed.

The third layer in the system architecture is a component which establishes a connection with database and creates web services which can be understood by third party software. Geoserver is a platform that can easily establish a connection with PostgreSQL and provide a Web Feature Service (WFS, OGC, 2014) for required data. A WFS enables storing geospatial and other non-spatial data associated to them. However, Geoserver can be used predominantly for manipulation of 2D data and simple 3D data. For more complex geospatial data, structuring data using json is not very efficient, requiring more time for streaming and drawing in third party software. For this reason, py3dtiles library was used which stores obtained data as chunks of binary data for quick streaming and drawing. Another reason for selecting this library is that it supports the creation of 3dtiles, a native format for visualisation of complex 3D geometries by the Cesium platform. Regarding point clouds, LOPoCS server enables storing point clouds to the database and streaming them to Cesium (LOPoCS, 2019). Therefore, all previously stored data can be obtained from the database and processed for visualisation.

The fourth and last component of the system architecture is visualisation of required geospatial data. In this regard, The Cesium platform was selected considering visualisation support for different 2D and 3D geospatial data, data styling and animation of time series data and web-based, open-source functionality. This component was included specifically for users of the system like UNSW students and employees who can access the system through any web browser and start interacting with the data immediately with no need for software deployment to potential customers.

## Data pre-processing

These section presents the 3D reconstruction of simple buildings (LOD1 according to CityGML), streets, gardens, their integration with a Digital Terrain Model (DTM) and the procedure for georeferencing BIM models.

### 3D object reconstruction

We used the footprint of buildings and DTM as input data. If the data are obtained from different sources, they must be converted to the same coordinate system. Then, the process to obtain topologically valid 3D objects and terrain, consists of four steps as follows (Fig. 2):

- Project building footprints to the terrain;
- Set height and create wall surfaces of the 3D buildings;
- Create roof and ground surfaces to create 3D building (solids);
- Recompute terrain considering building footprints as constraints.

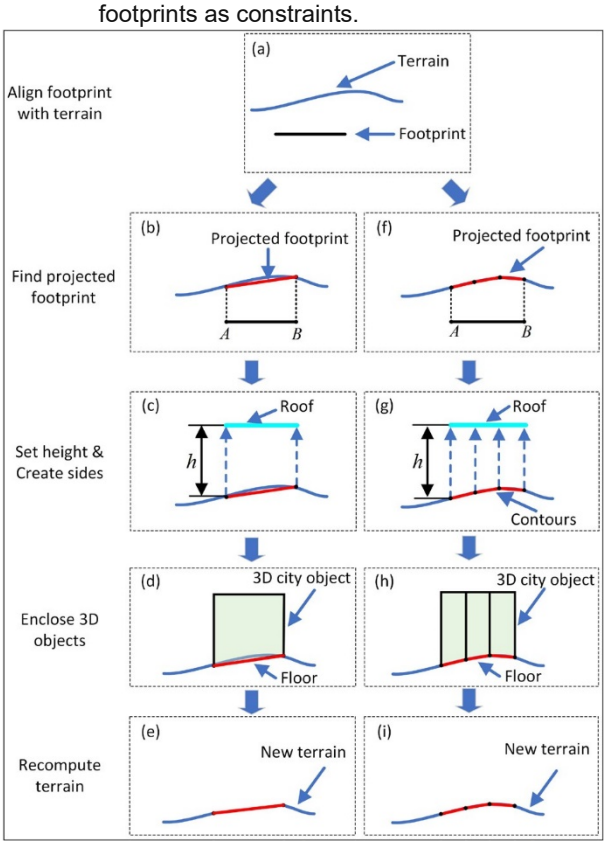



Figure 2 The process of 3D modelling based on footprint; a) projecting vertices only; b) draping the footprint onto the terrain

**Step 1: Find projected building footprints**

The footprint is normally presented as a 2D polyline, whose vertices are coplanar. Therefore, the first step is to find corresponding building footprints considering the terrain heights. Here we name corresponding building footprints as projected footprints. To make sure the final 3D building objects can be visualised correctly, their orientation is checked to be counter-clockwise to make sure the roof(s) and side(s) of final 3D building (solids) has the correct outward facing normal. However, a valid solid requires the ground surface to have also outward facing normal, therefore it must be flipped to be clock-wise oriented (Fig. 3).

Two methods are presented in this step. The first method is Projecting Vertices of footprints on the Terrain (PVT); the second is Projecting the whole footprint Polyline on the Terrain (PPT). The first method ensures the number of points of original footprints is preserved (Fig 2a). The second method ensures the curvature of the terrain is considered (Fig 2b). The first approach results in less points and therefore less facets on the walls, but the terrain is changed. The second approach might lead to unnecessarily complicated walls, which differ from reality. However, the terrain is preserved, which might be critical for some applications.

Naturally, the top view of projected footprints in the above two cases are the same as the top view of original footprints, see (b) and (f) in Fig. 2, and Fig. 4.

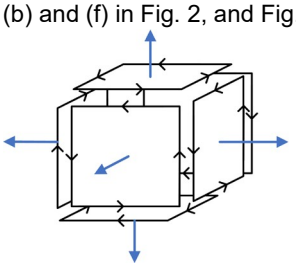


Figure 3 The outward facing normal of roof(s), side(s), and floor of an 3D building solid

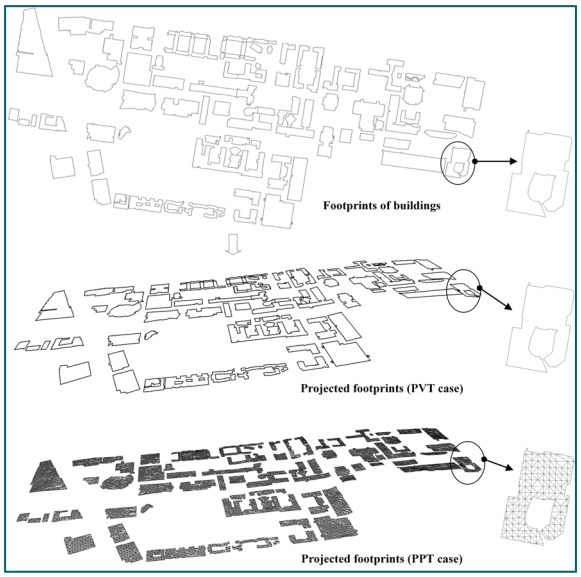



Figure 4 The projected footprints on the terrain

**Step 2: Set height and create wall surfaces**

After getting the ordered vertices of each projected footprint, walls of buildings are computed by extruding them up along the Z-direction based on *h*, see (c) and (g) in Fig. 2. Suppose one line segment of a polyline projected footprint, which has the start and end vertices $P0(x_0,y_0,z_0)$ and $P1(x_1,y_1,z_1)$ respectively. $P_0$ and $P_1$ will be extruded up along the Z-direction based on *h* to get $P_0'(x_0,y_0,h)$ and $P_1'(x_1,y_1,h)$. Then, these four vertices can form a wall, which can be represent as a *polygon* $(P_0, P_1, P_1', P_0', P_0)$.

As mentioned previously, the two different projection

approaches in Fig. 2 would results in two different results, for the PVT case, one polygon is a side, while for the PPT case, several polygons are combined to represent a side, see (d) and (h) in Fig. 2.

**Step 3: Create roof and ground surfaces to create 3D building (solids)**

The third step is to enclose side boundaries as 3D spaces (volumes) by adding roof and floor. All the roofs of the buildings are flat, and they are computed by making polygon surface from a set of vertices.

For the PVT case, the floors of can be created by connecting the projected vertices as 3D polyline and patching the polyline as a surface. The computation of the floor for the PPT case is more complicated than the other case. All projected points from one footprint belong to a group. We then deconstruct the projected footprint mesh into triangulations, then extract all edges (lines). By comparing the number of occurrences, the edges will be chosen to re-connect in order to form a contour. It should be noted that all vertices from one floor should be ordered in a clockwise direction before constructing the floor.

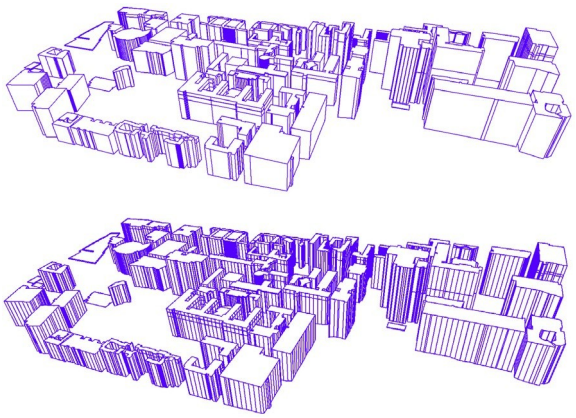
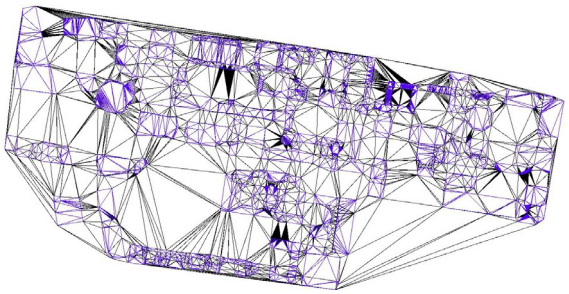


Figure 5 The created 3D buildings; (top) PVT method; (bottom) PVT method

**Step 4: Recompute terrain considering building footprints are constraints**

To keep the consistency between 3D object and terrain, the DTM must be re-computed considering the vertices and edges of projected building footprints as constraints (Fig. 2 e,f). In particular, the re-computation process is taking the vertices of projected footprints as *Points*, and their edges as *Breaklines* (constraints) to calculate a Constrained Delaney Triangulation (CDT) to represent the terrain. The results of the re-triangulated DTM are shown in Fig. 6. The footprints are clearly recognisable within the triangulated surface.



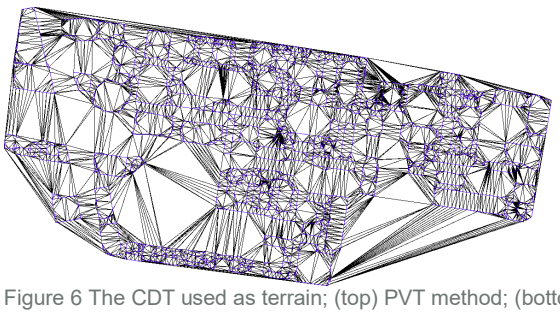


Figure 6 The CDT used as terrain; (top) PVT method; (bottom) PPT method

The same procedure is applied for 3D reconstruction of roads and gardens/lawns. The only difference is that these objects do not have height and therefore step 2 and 3 are not performed. Fig. 7 illustrates the results of the 3D road spaces.

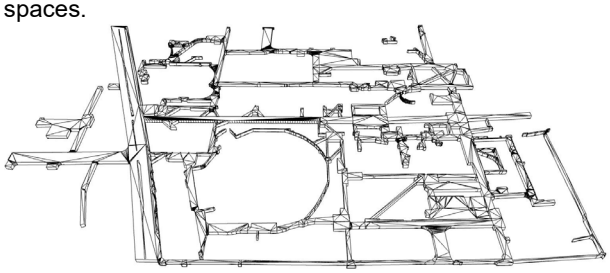


Figure 7 3D Road space reconstruction, where the height of the roads is 3 meters

Similarly, we have processed the trees, see Fig. 8. The dataset with trees consisted of 2D points. These points we projected on the terrain to get their real locations and the terrain was re-calculated as well. The trees are also managed in the database, see Fig. 9.

Figure 8 Georeferenced trees

| layer<br>character varying (254) | id<br>character varying (254) | code<br>character varying (254) | species<br>character varying (254) | geom<br>geometry |
|---|---|---|---|---|
| tree native | 2 | G325 | N | 01010000A0C46... |
| tree exotic | 2 | G279 | E | 01010000A0C46... |
| tree exotic | 13 | G260 | E | 01010000A0C46... |
| tree exotic | 8 | G461 | E | 01010000A0C46... |
| tree exotic | 3 | G27 NORTH | E | 01010000A0C46... |
| tree exotic | 1 | G186 | E | 01010000A0C46... |
| tree exotic | 1 | G188 | E | 01010000A0C46... |
| tree exotic | 3 | 257 | E | 01010000A0C46... |

Figure 9 Examples of trees in the table

## BIM pre-processing

The integration of BIM and GIS environments is known to be a challenging process (Liu et al., 2017). While GIS is suitable to represent geographic and urban features at a city scale, BIM models provide detailed representation at a building/construction level. The integration of both is therefore a critical combination for precinct modelling, as both city and building scales will present relevant aspects to consider. Therefore, in this project, available BIM models of the UNSW Campus were brought into the integrated GIS interface. More than just being able to visualise such models in their geographic context, one of the main challenges for us was to keep the rich information with BIM regarding

geometry and semantics of the building features. We could therefore successfully retrieve the geometry and semantic information from BIM models, store them in the database and query the necessary information from the GIS interface. Furthermore, we could also georeference the BIM model, which is considered one of the main limitations hindering BIM/GIS integration. Here, we will give more details about the whole process.

## From IFC to PostGIS

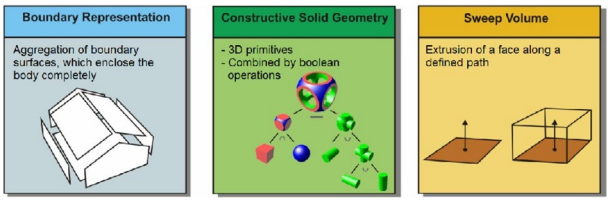



Figure 10 Types of geometry possible in an IFC file (Kolbe and Plumer, 2004)

BIM models are rich in geometry, semantic and topological information. Furthermore, the geometric information can come in different forms. In this part we are focussing on BIM models in the IFC format. In the latter, the geometry can be represented using Constructive Solid Geometry (CSG), Sweep volume or Boundary Representation (B-Rep).

Figure 10 illustrates the different geometry types. However, to bring those geometries into a GIS environment, only B-Rep should be used, which means that an explicit representation of the geometry in terms of points, edges and faces is necessary. Therefore, our approach consists of retrieving all the explicit geometries from the IFC model first, and then transforming those geometries into a GIS accepted format, while keeping the relevant semantic information.

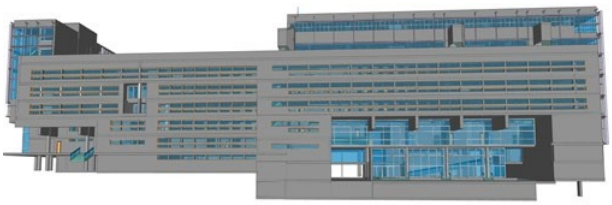


Figure 11 Red centre IFC model front view

Currently, most of the GIS software can handle 3D geometries. In PostGIS, a way to represent 3D geometries is to use spatial data types. One of these representations is human-readable and known as the Extended Well-Known Text (EWKT) representation, which is an extension of the Open Geospatial Consortium (OGC) WKT format that only supports 2D (PostGIS Doc., 2019). For example, a B-Rep volume from the IFC model can be represented as a POLYHEDRALSURFACE object and expressed in EWKT. The latter allows attaching a Spatial Reference IDentifier (SRID) to the described geometry. The *ST_GeomFromEWKT* function from PostGIS will then convert that object into a binary geometric object that is guaranteed to support 3D (no z- index will be dropped). Using POLYHEDRALSURFACE over other possibilities (e.g. MULTIPOLYGON Z) allows more PostGIS geometric functionality with the SFCGAL extension. We browse all the explicit geometries extracted at the loading of the file, collect them in the form of a SQL query, if the PostGIS extension is activated in the corresponding database, and we describe every polygon according to the EWKT standard. The query is then sent to the SQL server which will build the corresponding tables.

| | ifc_class character varying | ifc_guid character varying | ifc_name character varying | ifc_description character varying | ifc_containing_storey character varying | geom geometry |
|---|---|---|---|---|---|---|
| 1 | IfcSpace | 33Tvd9xin6NfBoj6oG0D_7 | 4101 | Room | Fourth -.52 | 01060000A0E610000020000000010300008... |
| 2 | IfcSpace | 23gGjrgOb1OPi5j592M8Q6 | 2046 | Room | Second+.52 | 01060000A0E610000002C0000000103000008... |
| 3 | IfcSpace | 240WIYRTn9fe9DomXtQjyL | G001 | Room | Ground | 01060000A0E610000030000000010300008... |
| 4 | IfcSpace | 23gGjrgOb1OPi5j592M8QI | 2Q12 | Room | Second+.52 | 01060000A0E6100000 1C0000000103000008... |
| 5 | IfcSpace | 23gGjrgOb1OPi5j592M8R6 | 2061 | Room | Second+.52 | 01060000A0E610000034000000010300008... |
| 6 | IfcSpace | 23gGjrgOb1OPi5j592M8R5 | 2062 | Room | Second+.52 | 01060000A0E610000034000000010300008... |
| 7 | IfcSpace | 3nUYDC4j99ARqekDGpELcg | M040 | Room | M2 | 01060000A0E610000044000000010300008... |
| 8 | IfcSpace | 3nUYDC4j99ARqekDGpELcI | M042 | Room | M2 | 01060000A0E610000024000000010300008... |
| 9 | IfcSpace | 3nUYDC4j99ARqekDGpELcl | MQ27 | Room | M2 | 01060000A0E6100000 1C0000000103000008... |
| 10 | IfcSpace | 3nUYDC4j99ARqekDGpELcX | M041 | Room | M2 | 01060000A0E610000028000000010300008... |
| 11 | IfcSpace | 3nUYDC4j99ARqekDGpELct | M046 | Room | M2 | 01060000A0E610000024000000010300008... |
| 12 | IfcSpace | 3nUYDC4j99ARqekDGpELcq | M043 | Room | M2 | 01060000A0E610000001C0000000103000008... |
| 13 | IfcSpace | 3nUYDC4j99ARqekDGpELb9 | M045 | Room | M2 | 01060000A0E6100000 1C0000000103000008... |

Figure 12 Example of tables created in the SQL database to store the IFC file of the Red Centre model

Along with the geometry, the information stored in the database includes the name of the IFC classes (e.g. IfcSpace, IfcWall, etc.), their unique IDs (corresponding to their *IfcGuid* attribute), their name which is an optional attribute in the IFC standard (e.g. component name and reference from the manufacturer, room number, etc.), their description (optional as well) and finally the name of the storey that contains each feature (see Figure 12).

Figure 13 Result of the direct import of the raw BIM geometry in a GIS platform (QGIS)

Thanks to the use of PostGIS, the stored geometry can be directly visualised in a GIS platform. For example, Figure 13 shows the IFC model of Figure 11, projected on an OSM map under the Pseudo-Mercator Coordinate Reference System (CRS, EPSG:3857). It can be seen that the scale of model is comparable to the biggest continents in size, and it lies very far away from where it would be expected to be. This is due to the lack of georeferencing information that should help to project the geometry of the BIM on the right CRS. We discuss in the next section how to address this problem.

## Georeferencing the BIM models

BIM models are coming from the Computer Aided Design (CAD) world and focus on the entities that they represent. Thus, when it comes to coordinate systems, their geometry is described in relation to the local coordinate system of the software used to generate them. For this reason, when directly imported in a GIS, the geometries of a BIM often end up somewhere in the middle of the ocean (see Figure 14) (Arroyo Ohori et al., 2018). The general method to solve this problem, is to manually select identified corresponding points between a georeferenced support and the un-georeferenced one, and then the adapted affine transformation is sought. In

this project. Because it is not always possible to find corresponding points easily, we adopted an automatic approach that allows projection of BIM geometries onto the CRS of a map.

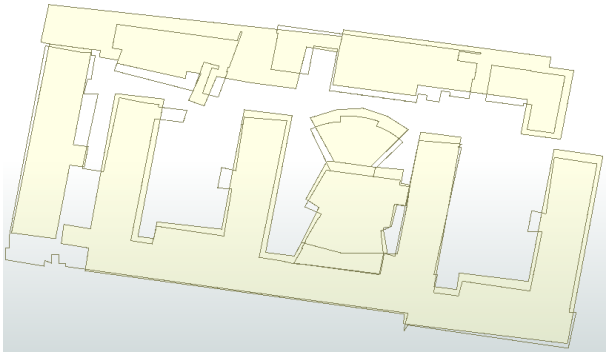


Figure 14 Footprint from EM (empty polygons) versus building projection from OSM (yellow polygons)

The method relays the 2D map information of the buildings that we want to georeference, such as footprint and projection, to the ground. Then the corresponding points from the similarly projected BIM are automatically detected, followed by computing the affine transformation that maps the projection of the BIM to the 2D feature. The input 2D georeferenced feature of the building plays a key role in the process. Figure 14 shows the difference between footprint data (empty polygons) and a building projection (yellow polygons) of the Red Centre (top) and the Old Main (bottom). While the footprints were provided by the Estate Management team (EM), the projections were obtained from Open Street Map (OSM). Although both data types are represented based on the WSG84 coordinate system (EPSG:4326) and projected on a Pseudo-Mercator map (EPSG:3857), the difference between them is caused by different projection methods. We assume that the data from EM is more accurate and primarily rely on it when possible. In this work, we have chosen to georeference the BIM model on the WSG84 CRS as well.

### Preparing the 2D data

We need to obtain data from the BIM models that is comparable with the data on the map in order to find the proper transformation to apply to them. The easiest way is to project the whole model on the xy-plane, but because BIM is an aggregation of several 3D objects, such direct projection will lead to complex features. It is not straightforward to obtain just the layout of the shell (see Figure 15). To overcome this, we use the 2D convex hull of the projected IFC geometries, as illustrated in Figure 15 (middle). Similarly, we computed the convex hulls of the available 2D features from the maps. Our assumption is that similar shapes will necessarily have similar convex hulls. Figure 15 (bottom) shows (in green) the convex hulls of the data from EM (which correspond to footprints) and (in blue) the one from the OSM data (which correspond to a global projection of the building).

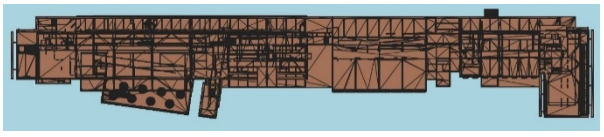



Figure 15 (Top) BIM geometries reprojected on the Pseudo-Mercator map. (Middle) Its corresponding convex hull. (Bottom) Convex hull of the EM footprints (green) and the one of the OSM building layout (blue)

It logically appears that, for the salmon-pink shape (convex hull of the projected IFC), the blue convex hull seems to be a better fit than the green one. This is because the way we computed the IFC convex hull is like the approach adopted by the OSM feature. Since only specific parts of the building are touching the ground in fact, retrieving the proper footprint would amount to retrieving specifically the features in the IFC model that are touching the ground. Unfortunately, there is no automatic and direct process to obtain such information. Therefore, we used the OSM feature for this case (the Red Centre model). However, this is not always the case for all models. For example, for the Round House, the 2D data from OSM clearly includes features that surround the original building.

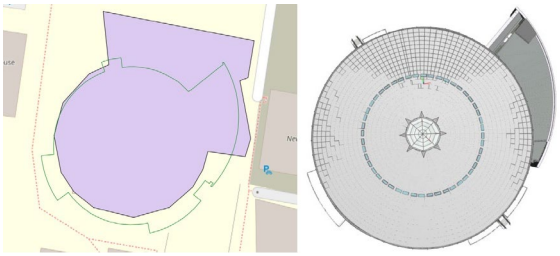


Figure 16 (Left) 2D data of the Round House including the footprint from EM (green/empty polygon) and the projection from OSM (purple polygon). (Right) Top view of the Round House IFC model (from a BIM viewer)

Figure 16 illustrates a case where the EM's footprint presents clearly a better match than the OSM's one, as it can be seen from the top view of the BIM model. Although the convex hulls between the map and the IFC look similar, they are still represented in different CRS. The coordinates (x,y) of the convex hull of the IFC are in meters while the ones from the map are in degrees (longitude, latitude). Since the transformation that we are seeking is affine, we converted the coordinates of the map features from degrees to meters. We used the projection from WSG84 (EPSG:4326) to the Pseudo-Mercator map (EPSG:3857), as it is commonly done for visualisation purpose on the web map services (including OSM, Google Maps, etc.).

- Finding the best alignment

Once a proper 2D feature is selected for the BIM model to geo-reference, and the coordinates of both BIM and map convex hulls are expressed in meters, the process of automatically finding the right mapping transformation can start. For this purpose, we implemented a combination of two approaches: the approximative rigid matching of Goodrich et

al. (1999) and the closed-form absolute orientation method of Horn (1987). The first method has the advantage to provide a good approximation of the alignment of the input shapes under Euclidean transformations (line and angle preserving translations and rotations) without requiring any point correspondence to be known between the input data. Furthermore, the sequence of points of the input shapes are not constrained to have the same cardinality, which are, combined with the first advantage, a powerful lever for the automation of the process.

Figure 17 (top) shows the result of the alignment between the convex hull of the projected IFC and the one of the OSM shape. Although the method of (Horn, 1987) has the disadvantage of requiring known corresponding points and same input size, it provides in one single closed-form step the exact best matching between the shapes and can handle the scaling as well. The latter parameter is very important as it can be seen that there is a difference of scale between the shape from the IFC and the one from the map. This difference results from the scale error involved in the WSG84 CRS (also replicated when projected on a Pseudo-Mercator) in comparison with the real meter distance from IFC.

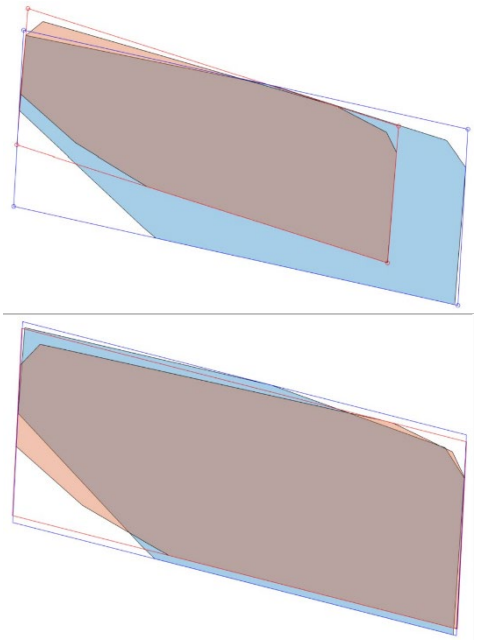


Figure 17 Alignment of the IFC (salmon-pink) and the OSM (blue) convex hulls. (Top) Alignment found after the approximative matching. (Bottom)Alignment after the absolute orientation

Thanks to the approximate alignment, we can rely on the oriented bounding boxes of the two shapes to retrieve four corresponding points between the shapes to align (3 points are the minimum necessary, but more points increase the precision of the alignment). The oriented corners of the boxes are then used as the input to the absolute orientation approach and the result is shown in Figure 15 (bottom). The shapes are not perfectly aligned, but now they have similar orientation and comparable scale.

- Applying the transformation to the IFC model


Figure 18 Georeferenced IFC model of the Red Centre. (Top) BIM model on top of the blue convex hull of the OSM feature used in the process. (Bottom) On top of the EM's footprint (green lines)

Now that we have the right affine transformation which comes in the form of a 4x4 matrix, we apply the transformation to all the coordinates of the IFC model. It is noteworthy to remind that all along the process the height of the IFC geometry is not altered. Figure 18 shows the result of the alignment of the BIM model on OSM (using QGIS), with, at its bottom the convex hull from the 2D OSM feature and the EM's footprint.

The result seems even better than the alignment of the convex hulls in Figure 16 (bottom). This can be explained by the fact that the convex hull provides a very generalised shape in comparison with the original one, and both convex hulls from the IFC and the map shapes may have tiny differences. Therefore, when the real geometry is placed on the map, the matching renders better as we have more details visible, yet there are still some inconsistencies.


Figure 19 The 3D georeferenced Red Centre model visualised on QGIS (with the DTM values used as height for the terrain)

The height of the IFC models is generally drawn consistently at model creation, with the assumption that z = 0 corresponds to the ground level of the building. But of course, this does not account for the height of the terrain under the model. To recover the right height values, we used the values of the DTM. For each building, we query the minimum DTM value $h_{min}$ that lies under its 2D shape. For every vertex of the transformed IFC model, $h_{min}$ is added to its z-coordinate. Figure 18 illustrates the final result of the georeferenced 3D IFC model of the Red Centre. The 3D geometries of all the BIM models are transformed and stored in the database following the same process.

# Data structuring

In this section a slightly simplified data model with respect to CityGML is described at the conceptual level using UML diagram. This diagram forms the basis for the implementation-dependent realisation of the model with a relational database system.

## 3D Modelling

With the proliferation of 3D city modelling applications, such as urban planning (Koninger et al. 1998, Knapps at al. 2007), navigation (Becker at al. 2007), disaster management (Zlatanovaet et al. 2004), environmental and training simulations (Randt et al. 2009), significant research efforts have been devoted efficiently and effectively determining an international standard of the Open Geospatial Consortium (OGC) for the representation and the exchange of 3D city models with regard to geometry, topology and semantics.

### Overview of CityGML

City Geography Markup Language (CityGML, cf. Groger et al. 2012) is an open data model using XML-based formats for the storage and exchange of virtual 3D city models. Due to its different, well-defined Levels-of-Detail (LOD) definitions, CityGML is employed for virtual 3D precinct modelling to address various complex GIS simulation and analysis tasks that go beyond pure 3D visualisation. It is explained in detail in the CityGML specification (Groger et al. 2008, Groger et al. 2012 and Kolbe 2009).

### CityGML and IFC

3D modelling is not only investigated in the context of geoinformation systems. The field of architecture, engineering, construction, and facility management (AEC/FM) as well as the field of computer graphics provide their own standards for the representation and exchange of 3D models. Building Information Modelling (BIM) means the semantic modelling of objects and processes in the field of AEC/FM and CAAD. As in CityGML, thematic objects are represented with their 3D spatial properties and interrelationships. Data are typically exchanged using the Industry Foundation Classes (IFC), an ISO standard describing a product model and data exchange format for the built environment developed by the International Alliance for Interoperability (IAI).

IFC provides a detailed semantic model for 3D building representations using constructive elements like beams, walls etc. Like in GML, IFC geometries are spatial properties of semantic objects. IFC has a very flexible geometry model (CSG, BRep and Sweep representations), but does not provide support for CRS. Since the scope of IFC is restricted to buildings and sites, no topographic feature classes like terrain, vegetation, water bodies etc. are included. IFC is a semantic model like CityGML, but with a different scope and at a different scale. IFC objects can be brought into the context of a city model within a GIS or spatial database and could then become subject to spatial and thematic queries.

### UML class diagram

Design decisions in the model are explicitly visualised within the UML diagrams. Following categories are presented in an integrated model: Building, City furniture, Terrain, Transportation, Vegetation and Land use. For detail, refer to data modelling report (Li et al., 2019)

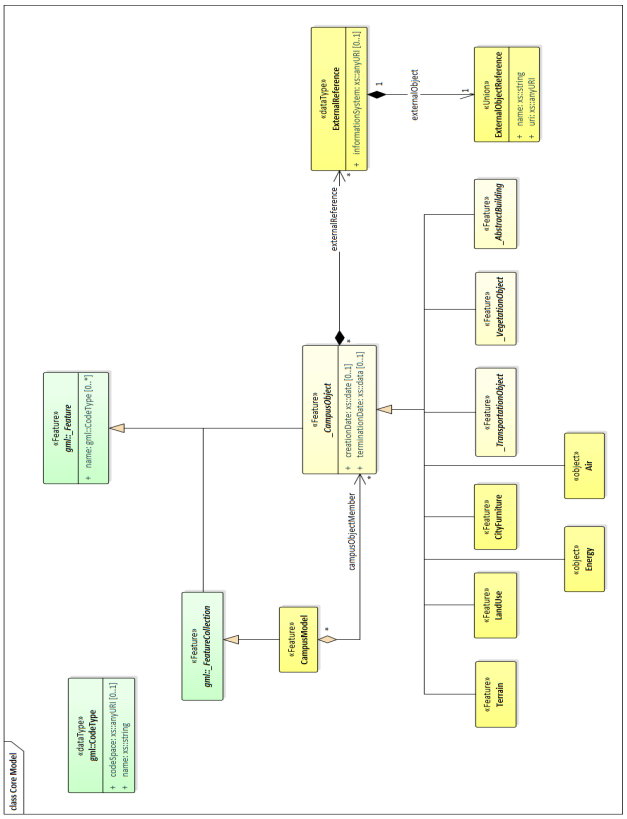



Figure 20 UML diagram of UNSW Campus (more details about the model can be found in Report on UML)

The thematic model consists of the class definitions for the most important types of objects within virtual 3D city models. Most thematic classes are derived from the basic classes *Feature* and *FeatureCollection*, the basic notions defined in ISO 19109 and GML3 for the representation of features and their aggregations. Features contain spatial as well as non-spatial attributes, which are mapped to GML3 feature properties with corresponding data types. The thematic model also comprises different types of interrelationships between Feature Classes like aggregations, generalisations, and associations.

The base class of all thematic classes within CityGML's data model is the abstract class *_CityObject*. As we have narrow down the scope of modelling to the UNSW campus, *_CityObject* will be replaced by *_UNSWCampusObject*, which provides a creation and a termination date for the management of histories of features as well as generic attributes and external references to corresponding objects in other datasets. Such a reference denotes the external information system and the unique identifier of the object in this system.

*_UNSWCampusObject* is a subclass of the GML class *Feature*, thus it may inherit multiple names from *Feature*, which may be optionally qualified by a *CodeSpace*. The generalisation property *generalizesTo* of *_UNSWCampusObject* may be used to relate features, which represent the same real-world object in different LoD.

Features of *_UNSWCampusObject* and its specialised subclass may be aggregated to a *UNSWCampusModel*, which is a feature collection with optional metadata. The subclasses of *_UNSWCampusObject* comprise the different thematic fields of a city model, in the following covered by separate thematic models: building model (*Building*), city furniture model (*CityFurniture*), digital terrain model (*Terrain*), land use model (*LandUse*), transportation model (*_TransportationObject*), vegetation model (*_VegetationObject*).

## Relational Database Schema

In this section, we give an overview of database solutions support the management of CityGML data. Since CityGML is a GML application schema, these software systems are able to automatically create database schemas for storing CityGML data for PostgreSQL/PostGIS, using the CityGML XML Schema definition files.

There are strong reasons to employ Spatially-extended Relational Database Management Systems (SRDBMS) to store and manage complex 3D city models. First, SRDBMS support all required geometry types and provide means for proper spatial indexing as well as for geometric and topological analyses. Second, SRDBMS can directly be used by most geoinformation systems (GIS) or spatially enabled ETL (Extract, Transform, Load) tools. There exists a variety of non-relational databases like object-oriented databases, which are increasingly investigated and employed in many application fields (cf. (Ordonez et al. 2010). However, they are currently still more or less limited in their capabilities and performance regarding spatial operations and coordinate transformations, which are of great importance for the enterprise use in GIS applications (cf. Agoub et al. 2016). Therefore, SRDBMS such as the commercial software ORACLE Spatial/Locator and the Open Source software PostgreSQL with PostGIS extension play a major role for GIS due to their extensive capabilities in handling 3D spatial data.

### Mapping rules, schema conventions

Generally, one or more classes of the UML diagram are mapped onto one table; the name of the table is identical to the class name (a leading underscore indicating an abstract class is left out). The scalar attributes of the classes become columns of the corresponding table with identical name.

### Multiplicities of attributes

Attributes with a variable number of occurrences (*) are substituted by a data type enabling the storage of arbitrary values – String. This means that object attributes can be stored in a single column.

### Cardinalities and types of relationships

n:m relations require an additional table in the database. This table consists of the primary keys of both elements' tables which form a composite primary key. If the relation can be restricted to 1:n or n:1 relationship the additional table can be avoided. Therefore, all n:m relations in CityGML were checked for a more restrictive definition. This results in simplified cardinalities and relations.

### Simplified treatment of recursions

Some recursive relations are used in the CityGML data model. Recursive database queries may cause high cost, especially if the number of recursive steps is unknown. In order to guarantee good performance, implementation of recursive associations receive two additional columns which contain the ID of the parent and of the root element.

The types of the attributes are customised to corresponding database (PostgreSQL/PostGIS) data type (see Table 1). Some attributes of the data type date were mapped to TIMESTAMP WITH TIME ZONE to allow a more accurate storage of time values.

Table 1 Data type mapping

| Data Type Mapping | |
|---|---|
| **UML** | **PostgreSQL/PostGIS** |
| **String, anyURI** | VARCHAR, TEXT |
| **Double** | NUMERIC |
| **Enumeration** | VARCHAR |
| **GML Geometry** | GEOMETRY |
| **CodeType** | VARCHAR |
| **Date** | DATE, TIMESTAMP WITH TIME ZONE |

### Table structure in PostGIS

Tables in PostGIS contain IFC objects and CityGML objects. Following tables are presented in our database server:

- Building
- Terrain
- Greensense
- Archibus
- Tree
- Lawn
- Bim
- Road
- MyAir

Table 2 Building table

| *Name* | *Data type* | *Length* | *Description* |
|---|---|---|---|
| **id** | INT | | primary key (PK) |
| **name** | VARCHAR | 200 | building name (FK) |
| **building_id** | INT | | building ID |

| | | | |
|---|---|---|---|
| **max_height** | REAL | | building max height |
| **geom** | geometry | | geometry |

Table 3 Terrain table

| *Name* | *Data type* | *Length* | *Description* |
|---|---|---|---|
| **id** | INT | | primary key (PK) |
| **vertex1_id** | INT | | vertex ID |
| **vertex2_id** | INT | | vertex ID |
| **vertex3_id** | INT | | vertex ID |
| **terrain_id** | INT | | terrain ID |
| **geom** | geometry | | geometry |

Table 4 Greensense table

| *Name* | *Data type* | *Length* | *Description* |
|---|---|---|---|
| **id** | INT | | primary key (PK) |
| **building** | VARCHAR | 200 | building name (FK) |
| **electricity_demand** | real | | Electricity Demand (kW) |
| **gas_flow_rate** | real | | Gas Flow Rate (m^3/s) |
| **timestamp** | TIMESTAMP | | Timestamp |

Table 5 Archibus table

| *Name* | *Data type* | *Length* | *Description* |
|---|---|---|---|
| **id** | INT | | primary key (PK) |
| **building** | VARCHAR | 200 | building name (FK) |
| **floor** | VARCHAR | 50 | floor name |
| **room** | VARCHAR | 50 | room name (FK) |
| **room_type_description** | VARCHAR | 200 | room type |
| **room_function_description** | VARCHAR | 200 | room function |
| **division_faculty_name** | VARCHAR | 200 | division faculty name |
| **unit_name** | VARCHAR | 200 | unit name |
| **current_room_capacity** | INT | | room capacity |
| **room_area_m_2** | NUMERIC | (5,2) | room area (m^2) |
| **room_comments** | VARCHAR | 200 | comments |

Table 6 Tree table

| *Name* | *Data type* | *Length* | *Description* |
|---|---|---|---|
| **id** | INT | | primary key (PK) |
| **layer** | VARCHAR | 200 | |
| **tree_id** | VARCHAR | 200 | tree ID |
| **code** | VARCHAR | 200 | tree code |
| **species** | VARCHAR | 50 | species |
| **geom** | geometry | | geometry |

Table 7 Lawn table

| *Name* | *Data type* | *Length* | *Description* |
|---|---|---|---|
| **id** | INT | | primary key (PK) |
| **lawn_id** | INT | | lawn ID |
| **name** | VARCHAR | 200 | lawn name |
| **geom** | geometry | | geometry |

Table 8 Bim table

| *Name* | *Data type* | *Length* | *Description* |
|---|---|---|---|
| **id** | INT | | primary key (PK) |
| **building** | VARCHAR | 200 | building name (FK) |
| **ifc_class** | VARCHAR | 50 | class type |
| **ifc_guid** | VARCHAR | 200 | object guid |
| **ifc_name** | VARCHAR | 200 | object name |
| **ifc_description** | VARCHAR | 2000 | description |
| **Ifc_containing_storey** | VARCHAR | 200 | storey |
| **geom** | geometry | | geometry |

Table 9 Road table

| *Name* | *Data type* | *Length* | *Description* |
|---|---|---|---|
| **id** | INT | | primary key (PK) |
| **road_id** | INT | | road ID |
| **geom** | geometry | | geometry |

Table 10 Myair table

| *Name* | *Data type* | *Length* | *Description* |
|---|---|---|---|
| **id** | INT | | primary key (PK) |
| **room** | VARCHAR | 50 | room name (FK) |
| **myair_num** | INT | | myair number |
| **co2** | INT | | co2 emission |
| **timestamp** | TIMESTAMP | | timestamp |

## Query on above tables

Tables including terrain, road, tree, lawn, are independent relations, while we should make associations among building, bim, Archibus, MyAir, and Greensense.

Case-I: Matching room information with IFC.

```
SELECT room, room_type_description, room_function_description FROM archibus, bim WHERE bim.building=archibus.building AND archibus.room=bim.ifc_name AND bim.ifc_class='IfcSpace' AND bim.building='Red Center - H13';
```

| | room<br>character varying (50) | room_type_description<br>character varying (200) | room_function_description<br>character varying (200) |
|---|---|---|---|
| 1 | B034 | Office - Professional/General Staff | CENTRAL ADMINISTRATION |
| 2 | GQ05 | Circulation Space | UNASSIGNABLE |
| 3 | GQ14 | Circulation Space | UNASSIGNABLE |
| 4 | G025 | Other Plant Room | UNASSIGNABLE |
| 5 | MQ18 | Comms Network System/Telephone PABX | UNASSIGNABLE |

Case-II: Matching co2 emission with room information.

```
SELECT SUM(co2) FROM MyAir, bim WHERE MyAir.room=bim.ifc_name AND building='Red Center - H13' AND ifc_class='IfcSpace' AND MyAir.room='4036' AND '2018-12-03'<=timestamp AND '2018-12-04'>timestamp;
```

| | sum<br>bigint |
|---|---|
| 1 | 1142 |

# Data visualisation

Data visualisation is an important component of the system architecture, providing an option to users to interact with the data. In this regard, a web-based platform called Cesium is selected, which can be opened in any web browser. Cesium is currently the most popular freeware option for visualisation of geospatial 3D content on web. However, a connection between PostgreSQL and Cesium is also required. Thus, a middle layer between those two that plays an important role is covering aspects of web servers and their services. In the next following sections web servers utilised for this project and Cesium platform were explained.

## Web servers and services

Web servers enable connection to databases and web application. Their main purpose is to prepare data for fast manipulation and visualisation afterwards in a web application.

When it comes to web GIS servers, Geoserver is commonly used and available for more than a decade commercially. This server enables fast creation of a Web Feature Service (WFS), encompassing spatial and non-spatial information which can be read by many web GIS applications. However, the main issue comes with processing 3D data. The problem is related to the way of storing 3D spatial data, which is the same as for 2D data neglecting the complexity of some 3D data like BIM models. Thus, data needs much more time to be processed by web applications in distributed mode, in the same way as 2D data.

3D tiles were proposed as a possible solution. The main idea behind them is to create and represent complex geometries through chunks of data, which can be loaded separately in a web application. At the same time, 3D tiles store data into binary files for faster distribution and load, in contrast to human readable files like json files, which Geoserver creates.

3D tiles exist in two formats. The first can store multi polygons and solids (i.e., b3dm files), while the second one can save only point cloud data (i.e., pnts files). An open-source library that enables the creation of 3D tiles is py3dtiles, written in Python programming language. LOPoCS is a web server that can use this library and distribute data to a web application. This server enables streaming real-time pnts files to a web application. For b3dm files, there is no publicly available server that can manipulate and stream data. Considering that, b3dm files representing 3D geospatial data are just exported using the py3dtiles and stored on the server side where Cesium is running. The point clouds are streamed by LOPoCS server. Datasets exported as b3dm files are BIM modes, extruded buildings, terrain and pedestrian streets from the UNSW campus.

Geoserver is also used for distributing datasets which are represented as points in the database, like trees and for non-spatial data, like energy consumption data coming from Greensense and air quality data coming from MyAir.

## 3D Visualisation using Cesium

The interface for Cesium is a globe with several options for choosing a background map (Fig. 21). Users can select different maps as an underlay for their content. Cesium also provides an option to select a terrain, in which case “Cesium World Terrain” option provides to some extent a realistic representation of ground terrain in 3D (Fig. 22 (left)). In the following text, we explain how this option was selected for representation of 3D content.

A clock and time line located at the bottom of the interface are used to show animated changes through time (Fig. 23). In our case, we used it to show energy consumption for several buildings, and visualisation of air quality in some rooms. In the left-top corner, we positioned a panel with different datasets that users can show separately or together with other datasets. Thus, 13 different datasets were identified and presented, showing available datasets previously stored in a database (Fig. 22 (right)).


Figure 21 Cesium default scene

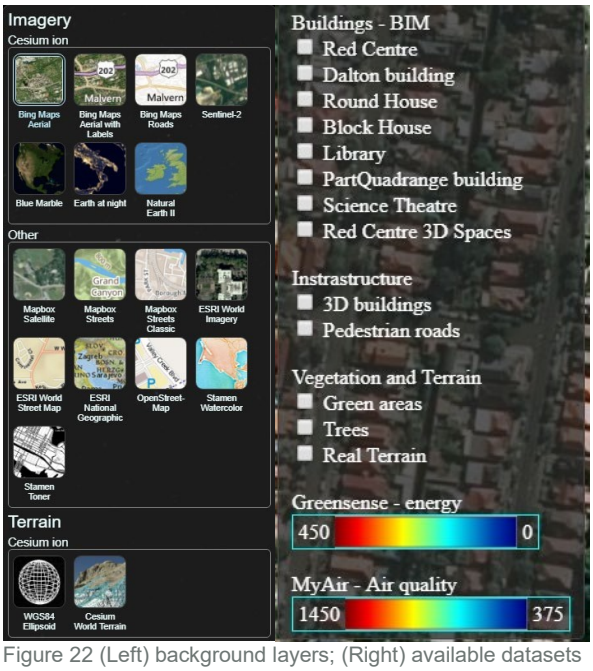



Figure 22 (Left) background layers; (Right) available datasets

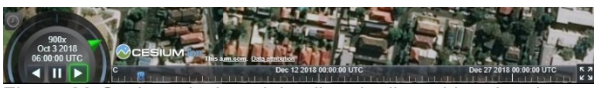


Figure 23 Cesium clock and timeline dealing with animations

## Terrain

Although Cesium provides a terrain to users showing height

changes, it is not fit for purpose. There is a difference of several meters with our terrain and it is inconsistent per area. Therefore, a terrain built from point clouds was fetched from the database and visualised. In this way a realistic, consistent and correct representation of the entire 3D environment was realised. All other content was aligned and based on the identified terrain (see Section 3).

### BIM Visualisation

As explained before, there are several ifc classes that where available in the BIMs provided by UNSW. Some of the models were more complete and correspondently heavier for transfer to Cesium, while others presented only the exterior of buildings. For example, the size of BIM of Round House building was 227 MB, while BIM of Dalton building was only 2 MB. BIM models' features were predominantly represented with fewer colours and no texture. Thus, a single colour for each feature was stored in the database, which was afterwards used for styling building components (Fig. 24). For six buildings BIM models were presented on Cesium platform (Fig. 25).

Figure 24 BIM model of the UNSW Library

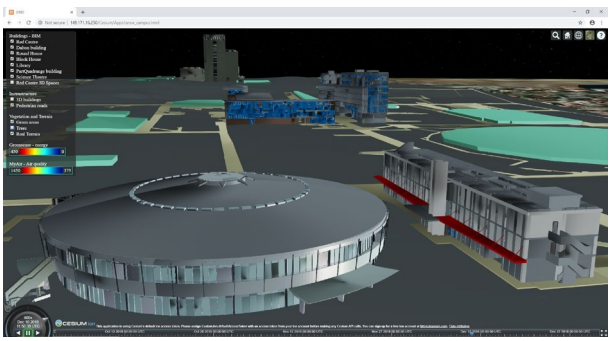



Figure 25 Visualisation of the BIM models in Cesium

### Extruded Buildings

Since BIM models were not available for every building at the UNSW campus, extruded building representations were created (see section 3) and used in Cesium as their replacement (Fig. 26). These buildings were also used for time-related visualisation of energy information, showing different colours based on a specific recording. For example, energy consumption for three months at regular intervals was visualised by animation. Content representing green space and pedestrian streets were also slightly extended in the third dimension for visualisation purposes.

Figure 26 Extruded buildings and BIM integration

Figure 27 Extruded buildings, along with pedestrian streets, green space and trees

### Trees

The dataset representing trees was visualised as 3D points. A request was sent to Geoserver creating a WFS out of data previously stored in the database, which was automatically visualised in Cesium as GeoJSON.

### Point cloud

The point clouds were streamed using LOPoCS server directly to Cesium. The amount of data needed to stream was approximately 1 million points, which needed around 30 seconds to display. The dataset represented a combination of LiDAR data and high-resolution images downloaded from Google Earth creating coloured point clouds (Fig. 28). We should mention that point cloud visualisation was only available on localhost, since only one port was available for remote connect to the server. Streaming more point clouds was also tested, but the server was showing some bugs requiring more testing and code debugging.

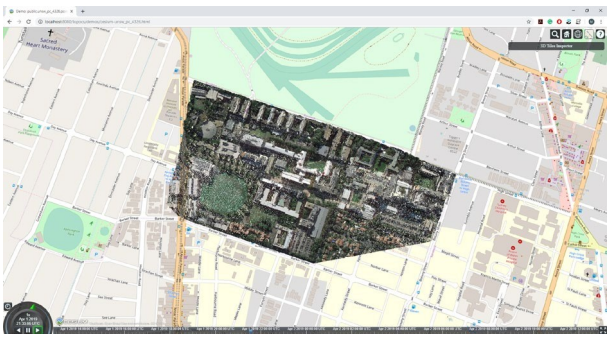


Figure 28 coloured point cloud visualisation in Cesium (~1 million points)

### Time-related data visualisation

There were two datasets representing changes through time for a specific parameter. Energy consumption was shown for 4 buildings considering a period of 3 months (i.e., October – December 2018) downloaded from Greensense system (Fig. 29). Based on this, the timeline at the bottom of the screen shows this period. The user could drag the time cursor left or right to a specific period. The time was sped up 900 times compared to real time, as the data were stored with timestamp of 15 minutes (i.e., 900 seconds) in the database. The styling is done based on the identified range for data between 0 and 450 as well as current value for energy consumption.

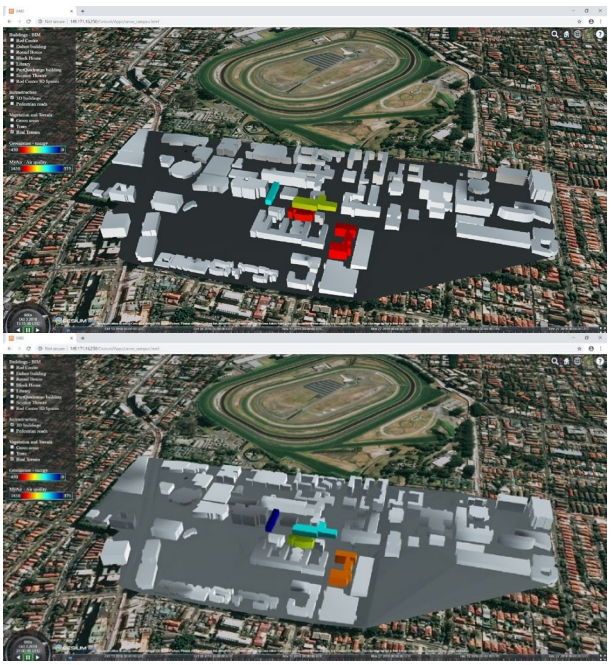


Figure 29 Energy consumption for 4 buildings at two different times

The timestamp was also 15 minutes for air quality data, received from the MyAir system. The data presented CO2 (i.e., carbon dioxide) coming from 54 sensors placed in the same number of rooms in Red Centre (West wing) building (Fig. 30). Data was only available for December 2018. Since the project was in the initial stage of establishment most of sensors did not have constant measurements, in which case rooms were presented as transparent objects. In other case, spaces representing rooms were styled based on identified minimum (375) and maximum (1450) values from the data as well as current measurement.

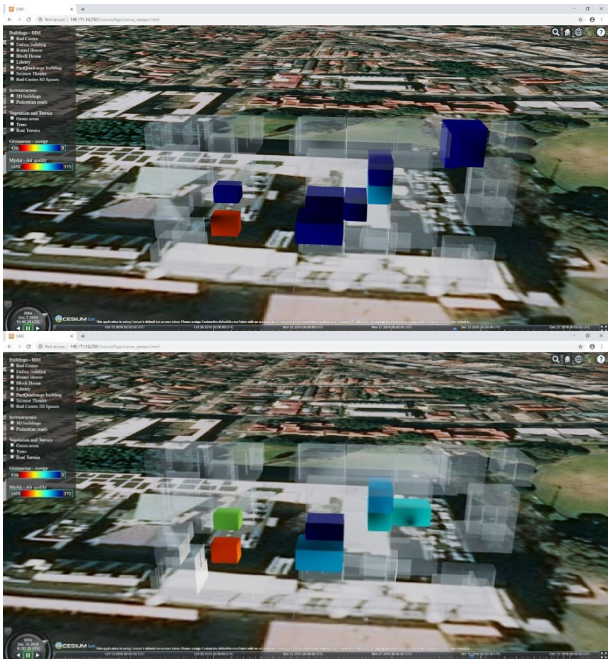


Figure 30 Air quality at two different times

### Attributes selection

Another aspect that should be mentioned is related to data manipulation. Apart from data visualisation and styling, feature properties can also be extracted and presented to users. Therefore, users can select each feature, highlighted with green colour, with left mouse click and in right top corner a panel would appear showing corresponded metadata. The two figures below show feature properties of a building and an ifc space representing one room in Red Centre. Although it has been developed, filtering and data manipulation based on these properties can be enabled easily.

## Discussion

This report presents the steps to create 3D models from different datasets, their organisation in a database management system (PostGIS) and their visualisation in Web-based environment (Cesium). The proposed system architecture, software tools and algorithms convergingly demonstrate that a large number of different datasets for a precinct area can be streamed, using international standards, into unified data structure. Such structuring allows access to the data from front-end software with standard components.
The entire process from data collection to visualisation has also exhibited a number of challenges. Many datasets are 2D and automatic procedures are needed to create 3D models. In this project we have created a 3D model from 2D footprints and an existing DTM. In this procedure the height of the buildings can come from different data, such as point clouds, OSM, or other 2D datasets.

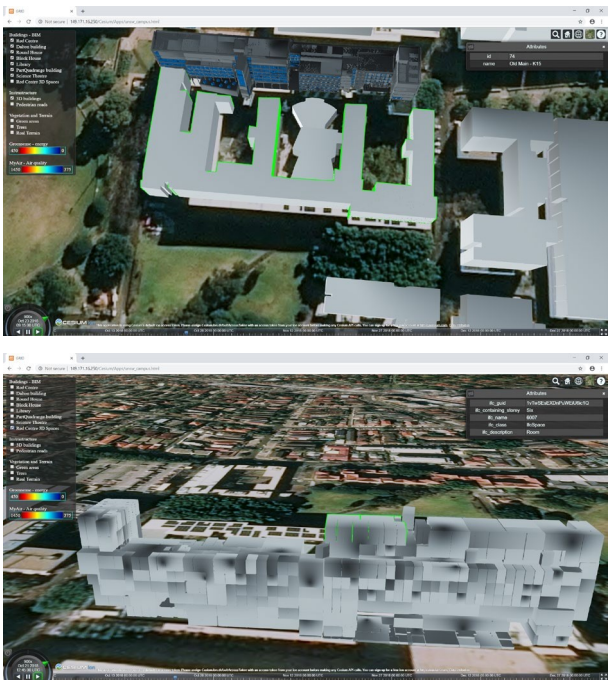


Figure 31 Feature selection and attributes presentation

Comparing the presented two projection methods, we can conclude that the details of building footprints and terrain are mutually influenced. Detailed footprints can lead to more complex terrain and vice versa. Deciding to keep detailed footprints or detailed terrain depends on the accuracy of the datasets and the applications. The proposed procedure ensures that topological consistency between buildings and terrain (no gaps, sinking or floating objects) is preserved.

Regarding the BIM models, we have shown an interesting approach to deal with them in a precinct model. Despite often over-rich information for GIS, storing information from BIM models in a database gives the possibilities to query and choose what information to view/analyse. This contributes to the flexibility and efficiency of the final model. Furthermore, our approach to store 3D geometries in the PostGIS database allows an efficient preservation of their properties and the automatic georeferencing to put them in their real spatial context. A future improvement would be to look at a higher connection between the BIM models and the other features (ground surface, terrain, etc.) to enable seamless analysis.

We gathered a deep understanding of the data structure of different data sources provided by UNSW, including spatial and non-spatial attributes. We investigated the two most used standards in GIS and BIM, i.e. CityGML and IFC. Based on this we created a generic conceptual 3D data model largely utilising CityGML concepts for outdoor in order to improve the information management paradigm to better support all process and activities in the UNSW campus. The main advantage of the conceptual model is that it facilitates data exchange, sharing of information and adding new concepts. The current conceptual model is based on the data inventory provided. New applications are posing new challenges. It is inefficient to start from scratch to insert/upgrade as new data is generated in the future. Thus, extendibility becomes an important requirement of our model. In addition, as the amount of data increases, we need to strengthen the scalability of our model. Another consideration when dealing with data from different sources is consistency- or dealing with inconsistency.

Regarding web servers and their services, there are several aspects that should be discussed. Geoserver represents a good platform to manipulate 2D and to some extent 3D content (e.g., points and lines). For more complex and heavier in size geometries, creating 3Dtiles is better option. In this regard, LOPoCS server provide an option to create real-time and stream 3Dtiles which can store point clouds (i.e., pnts files). However, it needs more work to ensure it is robust for manipulation with different datasets. The capabilities of the same server can be extended to support real-time streaming and manipulation of other 3dtile type (i.e., b3dm files). In this way, the server would effectively deal with BIM models and other more complex 3D geometries. Another aspect that should be explored is how to embed and store textures along with geometry models for photorealistic representation of buildings and environment.

In terms of data visualisation using the Cesium platform, several aspects should be discussed. Although 3dtiles allow asynchronised streaming of data into Cesium, the amount of content requires that the streaming should be taken into consideration each time.

Manipulation of data considering update of attributes and geometries should be further investigated. An interface should be developed in such a way that the database would be automatically updated with the changes if needed. Updates can be executed via other software packages such as QGIS, ArcGIS, FME, Revit, Bentley Systems and so forth.

Since PostGIS and many other extensions of PostgreSQL provide various capabilities for fast manipulation of geospatial data, many operations can be requested via the Cesium platform, executed on the database side and the results brought back to Cesium for visualisation.

# References


Agoub, A., Kunde, F. and Kada, M.A.R.T.I.N., 2016. Potential of graph databases in representing and enriching standardized Geodata. Tagungsband der, 36 (*pdf*)

Arroyo Ohori, K., Diakité, A., Krijnen, T., Ledoux, H. and Stoter, J., 2018. Processing BIM and GIS models in practice: experiences and recommendations from a GeoBIM project in The Netherlands. ISPRS International Journal of Geo-Information, 7(8), p.311 (*pdf*)

Becker, T., Nagel, C. and Kolbe, T.H., 2009. A multilayered space-event model for navigation in indoor spaces. In 3D geo-information sciences (pp. 61-77). Springer, Berlin, Heidelberg (*pdf*)

Goodrich, M.T., Mitchell, J.S. and Orletsky, M.W., 1999. Approximate geometric pattern matching under rigid motions. IEEE Transactions on Pattern Analysis and Machine Intelligence, 21(4), pp.371-379 (*pdf*)

Gröger, G., Kolbe, T.H., Czerwinski, A. and Nagel, C., 2008. OpenGIS city geography markup language (CityGML) encoding standard, version 1.0. 0 (*pdf*)

Gröger, G., Kolbe, T.H., Nagel, C. and Häfele, K.H., 2012. OGC city geography markup language (CityGML) encoding standard (*pdf*)

Horn, B.K., 1987. Closed-form solution of absolute orientation using unit quaternions. Josa a, 4(4), pp.629-642 (*pdf*)

Kolbe, T.H., 2009. Representing and exchanging 3D city models with CityGML. In 3D geo-information sciences (pp. 15-31). Springer, Berlin, Heidelberg (*pdf*)

Kolbe, T.H. and Plumer, L., 2004. Bridging the gap between gis and caad geometry, referencing, representations, standards and semantic modelling. GIM international, 18, pp.12-38

Köninger, A. and Bartel, S., 1998. 3D-GIS for urban purposes. Geoinformatica, 2(1), pp.79-103 (*pdf*)

Knapp, S. and Coors, V., 2007. The use of eParticipation systems in public participation: the VEPs example. In Urban and Regional Data Management (pp. 105-116). CRC Press (*link*)

Lapierre, A. and Cote, P., 2007, October. Using Open Web Services for urban data management: A testbed resulting from an OGC initiative for offering standard CAD/GIS/BIM services. In Urban and Regional Data Management. Annual Symposium of the Urban Data Management Society (pp. 381-393) (*pdf*)

Li, W., M. Aleksandrov, J. Barton, A. Diakité, J. Yan, S. Zlatanova, 2019, Progressing Precinct Modelling on the UNSW Campus and Beyond: BIM/PIM and 3DGIS- 3D PIM Modelling and Data Import, CRC Low Carbon Living report, 28p.

Liu, X., Wang, X., Wright, G., Cheng, J., Li, X. and Liu, R., 2017. A state-of-the-art review on the integration of Building Information Modeling (BIM) and Geographic Information System (GIS). ISPRS International Journal of Geo-Information, 6(2), p.53 (*pdf*)

OGC, 2014, Web Feature Service 2.0 Interface Standard, Open Geospatial Consortium, available at: http://docs.opengeospatial.org/is/09-025r2/09-025r2.html (last accessed March 2019)

Ordonez, C., Song, I.Y. and Garcia-Alvarado, C., 2010, October. Relational versus non-relational database systems for data warehousing. In Proceedings of the ACM 13th international workshop on Data warehousing and OLAP (pp. 67-68). ACM (*pdf*)

Randt, B., Bildstein, F. and Kolbe, T.H., 2007, July. Use of virtual 3d landscapes for emergency driver training. In Proc. of the Int. Conference on Visual Simulation IMAGE (Vol. 2) (*pdf*)

Zlatanova, S. and Holweg, D., 2004, March. 3D Geo-information in emergency response: a framework. In Proceedings of the 4th International Symposium on Mobile Mapping Technology (MMT'2004), March (pp. 29-31) (*pdf*)

Zlatanova, S., S. Shirowzhan, J. Yan, M. Aleksandrov, A. Diakité, J. Barton, 2019, Progressing Precinct Modelling on the UNSW Campus and Beyond: BIM/PIM and 3DGIS- Data Inventory, CRC Low Carbon Living report, 28p

Software

Archibus. (2019). Retrieved from https://archibus.unsw.edu.au/archibus/schema/ab- core/views/process-navigator/navigator-details.axvw

Cesium. (2019). Retrieved from https://cesiumjs.org/

CityData. (2019). Retrieved from https://citydata.be.unsw.edu.au/

CityGML. (2019). Retrieved from http://www.citygml.org/ Geoserver. (2019). Retrieved from http://geoserver.org/

Greensense. (2019). Retrieved from http://greensense.com.au/

IFC. (2019). Retrieved from https://en.wikipedia.org/wiki/Industry_Foundation_Classe

LOPoCS. (2019). Retrieved from https://github.com/Oslandia/lopocs

PostgreSQL. (2019). Retrieved from https://www.postgresql.org/

PostGIS Documentation. httpdbmanagement.h tml#EWKB_EWKT (Retrieved on 28/03/2019).

Py3dtile. (2019). Retrieved from https://github.com/Oslandia/py3dtilesELVIS,

Delivering High Resolution Elevation Data via FME Cloud and Amazon Web Services, NSW Spatial Services, YouTube, [Online] Available at https://www.youtube.com/watch?v=wF2k25NNVws, June 8 2017, accessed March 2019