3D BUILDING MODELING IN LOD2 USING THE CITYGML STANDARD

Dimitra Preka, Anastasios Doulamis

School of Rural and Surveying Engineering, National Technical University of Athens,
9 Iroon Polytechneiou str, 15780 Zografou, Greece - dimpreka@central.ntua.gr, adoulam@cs.ntua.gr

11th 3D Geoinfo Conference

KEY WORDS: CityGML, 3D Modeling, Buildings, LoD2, Database, PostgreSQL, 3DCityDB, SketchUp

ABSTRACT:

Over the last decade, scientific research has been increasingly focused on the third dimension in all fields and especially in sciences related to geographic information, the visualization of natural phenomena and the visualization of the complex urban reality. The field of 3D visualization has achieved rapid development and dynamic progress, especially in urban applications, while the technical restrictions on the use of 3D information tend to subside due to advancements in technology. A variety of 3D modeling techniques and standards has already been developed, as they gain more traction in a wide range of applications. Such a modern standard is the CityGML, which is open and allows for sharing and exchanging of 3D city models. Within the scope of this study, key issues for the 3D modeling of spatial objects and cities are considered and specifically the key elements and abilities of CityGML standard, which is used in order to produce a 3D model of 14 buildings that constitute a block at the municipality of Kaisariani, Athens, in Level of Detail 2 (LoD2), as well as the corresponding relational database. The proposed tool is based upon the 3DCityDB package in tandem with a geospatial database (PostgreSQL w/ PostGIS 2.0 extension). The latter allows for execution of complex queries regarding the spatial distribution of data. The system is implemented in order to facilitate a real-life scenario in a suburb of Athens.

1. INTRODUCTION

In recent years, 3D modeling of spatial objects and entire cities has become more and more necessary to a wide range of applications, such as those related to urban planning, 3D Cadastre and Smart Cities. Said applications are increasingly used by a number of cities and rural areas. This is a complex process, which comprises an approach of visualization of heterogeneous data, such as digital vector drawings, 2D and 3D virtual data, measurements, and combined use of different software programs. In recent decades, in the construction field particularly, the need to share and exchange of information led to the development of technology and applications, such as Building Information Modeling (BIM) and 3D models of geospatial information (El-Mekawy, 2010). These models define spatial objects with geometric and semantic representations. Industry Foundation Classes (IFC) and City Geography Markup Language (CityGML) are the best known semantic standards that are used by these 3D spatial models of the real world (El-Mekawy, 2010).

Nowadays, increasingly detailed 3D models are the cornerstone of many industries, like cadastral applications, construction, architecture, film and video games (Góźdź, Pachelski, Van Oosterom, Coors, 2014). This demand is met by an increasing number software programs, proprietary and otherwise, that enable the application of 3D modeling techniques.

1.1 3D City Modeling

Nowadays, a rapidly increasing number of companies create virtual 3D city models for use in various markets, such as urban planning, telecommunications, disaster/crisis management, 3D Cadastre, tourism, navigation, facilities management, environmental simulations and Smart Cities applications (Gröger, Kolbe, Czerwinski, Nagel, 2008). 3D city models are digital representations of the Earth's surface and the spatial objects that compose a city. In such models, the representation and the relationship between spatial objects should be clear and modeled (Stadler, Kolbe, 2007). Most efforts to model 3D cities focus on the representation of geometrical models while disregarding the models' semantic and topological parts. As a consequence, these parts cannot be used in GIS applications in which spatial queries, analysis tasks and exportation of spatial data are implemented, and are deprived of interoperability between different software packages and users. Given that the limited capability of models' reuse confines the wider use of 3D city models, a different approach to modeling had to be created for the purpose of covering the informative needs of many fields of study (Gröger, Kolbe, Czerwinski, Nagel, 2008).

In order for information from various applications to be reused, common standards should apply. In this regard, the CityGML model has been developed as a geospatial standard that comprises a semantic data model and an open standard. Consequently, it is suitable for all instances where urban objects can be represented and connected with various spatial relationships (Zhu, Li, Zhang, 2005).

Information modeling was originally created in the mid-1980s. The environment was greatly affected by the lack of communication between the various users and this had a negative impact on the efficiency and function of the industry. Research and Development (R&D) in this field has resulted in the development of BIM, in order to boost the construction area (El-Mekawy, 2010). BIM is a system that allows for 3D representation of construction and is a technological method where all relevant information, geometric and semantic, is contained in a 3D digital model.

Reference standards exist since 1988. R&D advancements resulted in the incorporation of the IFC standard in BIM applications, starting in 1996 (IAI, 1999). The IFC standard is not just about structural components' modeling, as it also represents various advanced procedures and analyses based on the spatial relationships between these components.

BIM and 3D models of geospatial information are currently considered a means for determining spatial objects with geometric and semantic representations. In a similar fashion, IFC and CityGML are the two best known semantic standards for the representation of designs and objects of the real world.

1.2 Previous Works on CityGML

For the storage and exchange of virtual 3D models of cities and landscapes the CityGML standard is used (Gröger, Kolbe, Czerwinski, Nagel, 2008) and (Kolbe, König, Nagel, Stadler, 2009). As well known, the standard is based on the Geography Markup Language 3 (GML3) schema (XML format) issued by the ISO TC211 and the Open Geospatial Consortium (OGC) (Gröger, Kolbe, Czerwinski, Nagel, 2008). CityGML includes generalization hierarchies between thematic classes, aggregations, relations between spatial objects and spatial properties and covers the geometrical, topological, semantic and appearance aspects of 3D city models. In addition, it differentiates between five consecutive Levels of Detail (LoD) (Gröger, Kolbe, Czerwinski, Nagel, 2008).

1.3 Contribution

The main goal of this paper is to develop an interoperable system able to properly depict, manage and handle complex 3D cadastre information of residential houses. The tool is built around the 3DCityDB package in relation with a geospatial database, PostgreSQL with the PostGIS extension (Kunde, Asche, Kolbe, Nagel, Herreruela, König, 2013). The latter allows for data organization and the implementation of complex queries regarding the topological relationships between features. This study is a first attempt at the implementation of an application capable of facilitating engineers who operate within the land management, registry and other relevant fields, offering appropriate data organization plus an easy and familiar interface. The solution was tested and proven in a real-life application in the suburb of Kaisariani, Athens.

2. METHODOLOGY

2.1 Data and Software

The main object of this research is the study and application of a technique of 3D modeling and visualization in LoD2 of a block of 14 buildings using the CityGML standard. The study area is located in Kaisariani, a municipality of Athens. For this study, three software packages were used: SketchUp, PosgtreSQL/PostGIS and 3D City Database (3DCityDB).

2.2 Procedures

Collected data, for the buildings of the study area, consists of their address, use, roof type, height and number of storeys and are to be recorded in a relational database.

The exterior of the buildings of the study area was designed in LoD2, in which the buildings' surfaces are divided into ground, wall and roof within SketchUp, by applying textures to the buildings' surfaces. As known in SketchUp, it is possible to define a georeference by importing in the designing environment of the program an appropriate background image from Google Earth (see Figure 1).

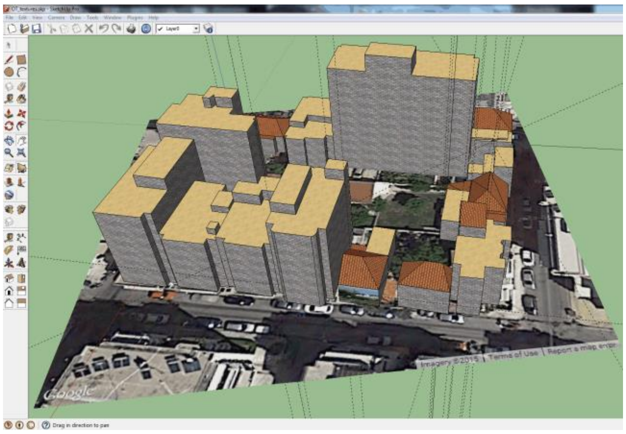

Figure 1. 3D Design of the Buildings in SketchUp

A CityGML file in GML3 schema of the 3D design followed by a folder with images of the used textures were exported by SketchUp too through the free plug-in, CityGML Editor 18 (see Figure 2). The information about the geometry and all spatial relations was derived by a 3D design of the buildings in .dxf format, provided by our Laboratory.

```
<core:cityObjectMember>
<bldg:Building>
<bldg:boundedBy>
<bldg:WallSurface>
<bldg:lod2MultiSurface>
<gml:MultiSurface>
<gml:surfaceMember>
<gml:Polygon gml:id="_OT_textures_BD.1_PG.2">
<gml:exterior>
<gml:LinearRing gml:id="_OT_textures_BD.1_PG.2_LR.1">
<gml:posList srsDimension="3">
742093.711821386 4206061.08817771 -1.67216250195015
742092.211254083 4206054.18457545 -1.67216250195015
742092.211254083 4206054.18457545 2.72970656660888
742093.711821386 4206061.08817771 2.72970656660888
742093.711821386 4206061.08817771 -1.67216250195015
</gml:posList>
</gml:LinearRing>
</gml:exterior>
</gml:Polygon>
</gml:surfaceMember>
</gml:MultiSurface>
</bldg:lod2MultiSurface>
</bldg:WallSurface>
</bldg:boundedBy>
```

Figure 2. Part of the Exported CityGML File by SketchUp

At the same time, a new empty database was created in PostgreSQL/PostGIS where the Coordinate Reference System (CRS) of the study area was defined and its schema was structured in the format of 3DCityDB using the Command Prompt/cmd.exe (see Figure 3). This schema includes 45 entities/tables (see Figure 5).

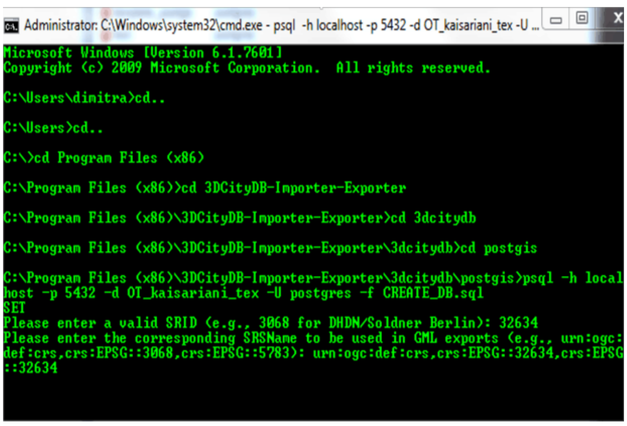

Figure 3. Creation of Database Structure in 3DCityDB

Subsequently, the empty database was populated automatically, but fine-tuned manually, with the aforementioned data types, within the 3DCityDB package and via the exported CityGML file from the first design step.

Prior to the exportation of the desired (final) files, certain parameters were defined in the 3DCityDB package, with regard to the appearance of the buildings in Google Earth. The extra tool selected for the visualization of the descriptive information of buildings is the "Balloon". Upon selection of a visualized building, a pop-up balloon with all pertinent information from the database, appears next to it (see Figure 4).

Figure 4. Balloon of a 3D Building in Google Earth

The application is complete with the exportation of a new CityGML file that includes building features and KML/COLLADA files that allow for visualization in Google Earth environment.

2.3 Data Input Process

The database in 3DCityDB format is compatible with the CityGML standard. Its schema called "public" is enriched with 45 tables which refer to a city in any LoD that it may be configured in (see Figure 5).

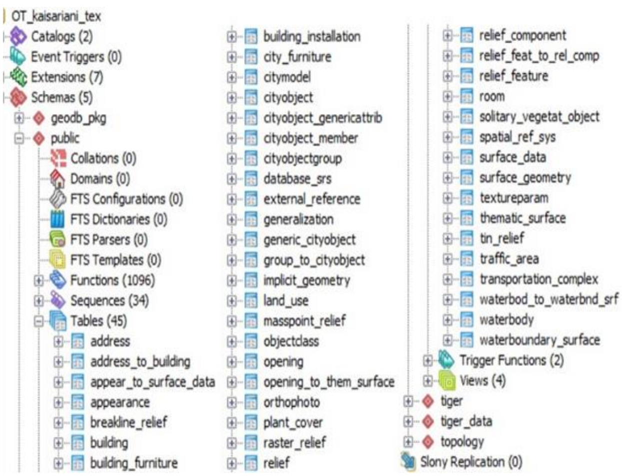

Figure 5. Schema Structure of 3DCityDB

In this application, the level of detail of the modeled study area is the LoD2, hence only the buildings' shells were modeled. Therefore, the database tables, which were used and processed, were limited and related only to buildings and in particular, to their exterior and location data and not to other thematic classes.

A key tool for the connection and communication between the 3DCityDB and the PostgreSQL database, the organisation of spatial and descriptive data in the database and the exportation of the final CityGML and KML/COLLADA files, is the "3DCityDB-Importer-Exporter" of the library of 3DCityDB package (see Figure 6).

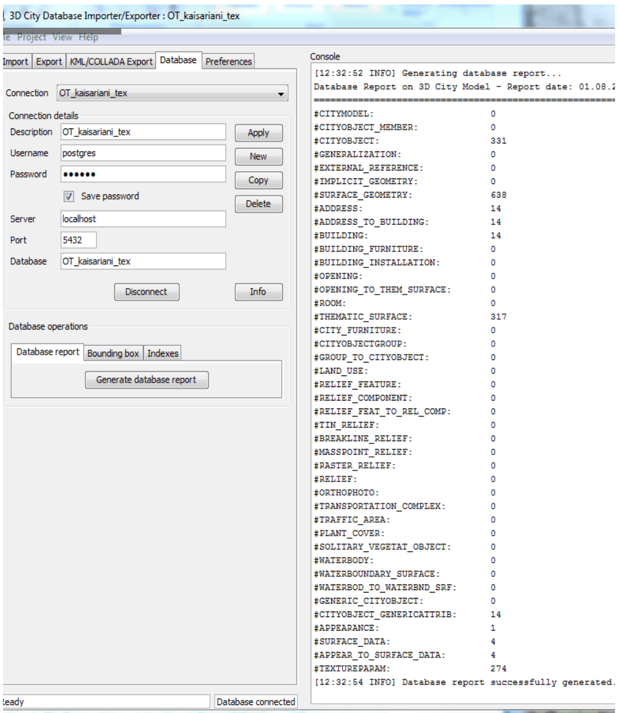

Figure 6. Environment of 3DCityDB-Importer-Exporter

After implementing the initial connection between PostgreSQL and 3DCityDB, the exported CityGML file from SketchUp, which includes all the spatial information, geometry and location of the buildings, as they arise from the 3D design in SketchUp, was inserted in 3DCityDB-Importer-Exporter. All the geometrical and topological information of the CityGML file was automatically registered in the appropriate fields of the respective database tables through the aforementioned key tool. For example, two tables in which this information was automatically registered is the table called "database_srs" (see Figure 7), which refers to the CRS/SRID of the study area and the table called "cityobject" (see Figure 8), which refers to each of the 14 buildings of the study area and to each of the surfaces that compose them. Each line refers to an object and contains the code of the category "class_id" from the table "objectclass", a coded name "gmlid", the geometric spread of "envelope geometry" in coded form, the creation date and modification and the user, who updates them.

	srid [PK] integer	gml_srs_name character varying(1000)
1	32634	urn:ogc:def:crs,crs:EPSG::32634,crs:EPSG::32634
*		

Figure 7. "database_srs" Table of 3DCityDB

Data - PostgreSQL 9.3 (localhost:5432) - OT_kaisariani_tex - cityobject

id [PK] serial	class_id integer	gmlid character varying	gmlid_codespace character varying	envelope geometry(PolygonZ,32634)	creation_date date	termination_ date	last_modification_date date	updating_ character
1	26	UUID_8a186ba5-3	UUID	01030000A07A7F000001000	2015-06-16		2015-06-16	postgres
2	26	UUID_402e8f5f-1	UUID	01030000A07A7F000001000	2015-06-16		2015-06-16	postgres
3	26	UUID_1566ab08-5	UUID	01030000A07A7F000001000	2015-06-16		2015-06-16	postgres
4	26	UUID_4528b883-e	UUID	01030000A07A7F000001000	2015-06-16		2015-06-16	postgres
5	34	UUID_f3ec754d-c	UUID	01030000A07A7F000001000	2015-06-16		2015-06-16	postgres
6	34	UUID_b1102ae8-8	UUID	01030000A07A7F000001000	2015-06-16		2015-06-16	postgres
7	34	UUID_cc909009-c	UUID	01030000A07A7F000001000	2015-06-16		2015-06-16	postgres
8	34	UUID_de3fb1a0-9	UUID	01030000A07A7F000001000	2015-06-16		2015-06-16	postgres
9	34	UUID_5f117dec-c	UUID	01030000A07A7F000001000	2015-06-16		2015-06-16	postgres
10	34	UUID_fc114f2c-7	UUID	01030000A07A7F000001000	2015-06-16		2015-06-16	postgres
11	34	UUID_51799ff5-6	UUID	01030000A07A7F000001000	2015-06-16		2015-06-16	postgres
12	34	UUID_47ef8ab9-b	UUID	01030000A07A7F000001000	2015-06-16		2015-06-16	postgres
13	34	UUID_0ebaed22-a	UUID	01030000A07A7F000001000	2015-06-16		2015-06-16	postgres
14	34	UUID_eed74cb9-8	UUID	01030000A07A7F000001000	2015-06-16		2015-06-16	postgres
15	34	UUID_d0fa11f2-7	UUID	01030000A07A7F000001000	2015-06-16		2015-06-16	postgres
16	34	UUID_1f1113ee-3	UUID	01030000A07A7F000001000	2015-06-16		2015-06-16	postgres
17	34	UUID_e038d760-8	UUID	01030000A07A7F000001000	2015-06-16		2015-06-16	postgres
18	34	UUID_160d8ea7-c	UUID	01030000A07A7F000001000	2015-06-16		2015-06-16	postgres
19	34	UUID_98edb8d9-7	UUID	01030000A07A7F000001000	2015-06-16		2015-06-16	postgres
20	34	UUID_606b22a4-0	UUID	01030000A07A7F000001000	2015-06-16		2015-06-16	postgres
21	34	UUID_6488b26c-9	UUID	01030000A07A7F000001000	2015-06-16		2015-06-16	postgres
22	34	UUID_997413ca-1	UUID	01030000A07A7F000001000	2015-06-16		2015-06-16	postgres
23	34	UUID_a23c9ac3-1	UUID	01030000A07A7F000001000	2015-06-16		2015-06-16	postgres
24	34	UUID_1f9543d6-9	UUID	01030000A07A7F000001000	2015-06-16		2015-06-16	postgres
25	33	UUID_5a2b0af6-a	UUID	01030000A07A7F000001000	2015-06-16		2015-06-16	postgres

Figure 8. Part of the “cityobject” Table of 3DCityDB

In addition to the spatial information about the 14 buildings of the study area, their descriptive information was imported into the database as well. The descriptive information for each building was added manually in the corresponding fields of the database. This information refers to the addresses of the modeled buildings in the table called “address” (see Figure 9) and various features of the buildings, such as description, usage, roof type, measured height and storeys above the ground in the table called “building” (see Figure 10).

Data - PostgreSQL 9.3 (localhost:5432) - OT_kaisariani_tex - address

id [PK] serial	street character varying(1000)	house_number character varyi	po_box character vai	zip_code character vai	city character vai	state character vai	country character
1	Vrioulon	12		161 21	Kaisariani	Attiki	Greece
2	Vrioulon	14		161 21	Kaisariani	Attiki	Greece
3	Vrioulon	16-18		161 21	Kaisariani	Attiki	Greece
4	Smirnis	21		161 21	Kaisariani	Attiki	Greece
5	Smirnis	19		161 21	Kaisariani	Attiki	Greece
6	Iroon Politexniou	11		161 21	Kaisariani	Attiki	Greece
7	Iroon Politexniou	9		161 21	Kaisariani	Attiki	Greece
8	Iroon Politexniou	7		161 21	Kaisariani	Attiki	Greece
9	Iroon Politexniou	5		161 21	Kaisariani	Attiki	Greece
10	Iroon Politexniou	3		161 21	Kaisariani	Attiki	Greece
11	Iroon Politexniou	1		161 21	Kaisariani	Attiki	Greece
12	Tzon Kennenti	16		161 21	Kaisariani	Attiki	Greece
13	Tzon Kennenti	18		161 21	Kaisariani	Attiki	Greece
14	Tzon Kennenti	20		161 21	Kaisariani	Attiki	Greece

Figure 9. “address” Table of 3DCityDB

- building

building_root integer	description character varyi	class character	function character vai	usage character varying(1000)	year_of_cons date	year_of_dem date	roof_type character varying(25	measured_he double precis	storeys_above numeric(8,0)
1	monokatoikia	1		katoikia			keramoskepis & pla	6.12	1
2	polikatoikia	1		katoikia			plakoskepis & kera	11.03	2
3	polikatoikia	3		katoikia			plakoskepis	9.19	3
4	monokatoikia	1		katoikia			keramoskepis & pla	6.21	1
56	polikatoikia	6		katoikia			plakoskepis	24.11	7
87	monokatoikia	1		katoikia			keramoskepis	6.69	1
92	monokatoikia	2		katoikia			plakoskepis & kera	10.9	2
121	polikatoikia	3		katoikia			plakoskepis	12.35	3
133	monokatoikia	1		katoikia			keramoskepis	6.39	1
179	polikatoikia	4		katoikia			plakoskepis	16.36	4
202	polikatoikia	3		katoikia			plakoskepis	11.41	3
240	polikatoikia	4		katoikia & plintirio			plakoskepis	16.67	4
254	polikatoikia	4		katoikia			plakoskepis	19.67	5
281	sinergio moto	2		sinergio moto			plakoskepis	7.1	2

Figure 10. “building” Table of 3DCityDB

After reconnecting the 3DCityDB to the renewed PostgreSQL database and setting the parameters in relation to the appearance of the modeled buildings in Google Earth, the final and fully updated from the database CityGML file and the KML/COLLADA files (visualization files of modeled buildings in Google Earth) were exported by 3DCityDB-Importer-Exporter.

2.4 Database Queries

The relational database PostgreSQL is accompanied by an important tool/extension, PostGIS, which enables spatial functions therein. PostGIS supports the use of special spatial operands such as distance, length, perimeter, area etc., in order to be able to draw up spatial queries and functions. In brief, it extends the PostgreSQL, for the purpose of being able to raise a query, store and manage spatial data. Furthermore, it supports mixed types of data, such as points, lines, polygons and a plethora of Coordinate Reference Systems.

All ordinary SQL operands can be applied for the syntax of simple queries. Most functions with spatial functionality, which are supported by PostGIS, start with the "ST" (Spatial Type) prefix. As expected, the legs of writing a query are three: the selection (SELECT) of the columns, from which information on the results are going to be extracted, the reference tables (FROM), from which the aforementioned columns are selected and the expression of various conditions (WHERE). Three different examples of spatial queries’ syntax to the database follow, as well as their results.

The first example of a spatial query on the PostgreSQL database is the request “Sort the database buildings with their addresses based on their respective areas in descending order” (see Figure 11). In this case the “ST_Area” spatial function was applied to the column "envelope" of the "cityobject" table, which refers to geometric spread of the modeled objects. Moreover, one of the conditions is the "class_id" of the "cityobject" table to be equal to the value "26" that corresponds to the object class “buildings”.

SQL Editor | Graphical Query Builder

revious queries

```
SELECT
  building.id,
  ST_Area (cityobject.envelope) AS area_m2,
  address.street,
  address.house_number,
  address.zip_code,
  address.city,
  address.state,
  address.country
FROM
  public.building,
  public.address_to_building,
  public.address,
  public.cityobject
WHERE
  building.id = address_to_building.building_id AND
  building.id = cityobject.id AND
  address_to_building.address_id = address.id AND
  cityobject.class_id = '26'
ORDER BY area_m2 DESC;
```

utput | Explain | Messages | History

id integer	area_m2 double precision	street character varying(1000)	house_number character varying(256)	zip_code character varying(256)	city character varying(256)	state character varying(256)	country characte
56	400.346472917367	Vrioulon	16-18	161 21	Kaisariani	Attiki	Greece
240	219.961784174097	Tzon Kennenti	16	161 21	Kaisariani	Attiki	Greece
254	201.654520459952	Tzon Kennenti	18	161 21	Kaisariani	Attiki	Greece
179	190.75414452183	Iroon Politexniou	5	161 21	Kaisariani	Attiki	Greece
92	186.152893206395	Smirnis	21	161 21	Kaisariani	Attiki	Greece
1	179.773348423205	Smirnis	19	161 21	Kaisariani	Attiki	Greece
202	162.162045990783	Iroon Politexniou	1	161 21	Kaisariani	Attiki	Greece
4	159.732647165985	Vrioulon	12	161 21	Kaisariani	Attiki	Greece
3	157.086428919361	Vrioulon	14	161 21	Kaisariani	Attiki	Greece
121	155.863543097628	Iroon Politexniou	3	161 21	Kaisariani	Attiki	Greece
2	128.871168825109	Iroon Politexniou	11	161 21	Kaisariani	Attiki	Greece
133	127.624474993111	Iroon Politexniou	7	161 21	Kaisariani	Attiki	Greece
281	87.5522463255078	Tzon Kennenti	20	161 21	Kaisariani	Attiki	Greece
87	55.8240081820629	Iroon Politexniou	9	161 21	Kaisariani	Attiki	Greece

Figure 11. Example #1 of Spatial Query on PostgreSQL Database

The second example of a spatial query is the question “Which buildings of the database are located at a distance less than 12m from the 3D point with coordinates (742110, 4206040, 3) and which are their addresses?” (see Figure 12). Likewise, the "ST_3DDistance" spatial function was applied to the column "envelope" of the "cityobject" table and to the 3D point of the query indicating the defined SRID (32634) of the database. The same condition of the previous query that refers to the "class_id" of the "cityobject" table to be equal to the value "26", was applied to this query as well.

L Editor | Graphical Query Builder

ous queries

```sql
SELECT
  building.id,
  address.street,
  address.house_number,
  address.zip_code,
  address.city,
  address.state,
  address.country
FROM
  public.building,
  public.address_to_building,
  public.address,
  public.cityobject
WHERE
  building.id = address_to_building.building_id AND
  building.id = cityobject.id AND
  address_to_building.address_id = address.id AND
  cityobject.class_id = '26' AND
  ST_3DDistance(cityobject.envelope, ST_GeomFromEWKT('SRID=32634;POINTZ(742110 4206040 3)')) < 12;
```

utput | Explain | Messages | History

id integer	street character varying(1000)	house_number character varying(256)	zip_code character varying(256)	city character varying(256)	state character varying(256)	country character v
4	Vrioulon	12	161 21	Kaisariani	Attiki	Greece
56	Vrioulon	16-18	161 21	Kaisariani	Attiki	Greece
3	Vrioulon	14	161 21	Kaisariani	Attiki	Greece
254	Tzon Kennenti	18	161 21	Kaisariani	Attiki	Greece

Figure 12. Example #2 of Spatial Query on PostgreSQL Database

The last example of a spatial query is the question “Which buildings of the database are overlapped with the framework with coordinates (742110, 4206040) on its northwest end and coordinates (742150, 4206061) on its southeast end and which are their addresses?” (see Figure 13). In this example, the function "BOX3D" was applied in addition to the condition that requires the "class_id" of the "cityobject" table to be equal to the value "26".

SQL Editor | Graphical Query Builder

Previous queries

```sql
SELECT
  building.id,
  address.street,
  address.house_number,
  address.zip_code,
  address.city,
  address.state,
  address.country
FROM
  public.building,
  public.address_to_building,
  public.address,
  public.cityobject
WHERE
  building.id = address_to_building.building_id AND
  building.id = cityobject.id AND
  address_to_building.address_id = address.id AND
  cityobject.class_id = '26' AND
  envelope && 'BOX3D(742110 4206040, 742150 4206061)'::box3d;
```

Output | Explain | Messages | History

id integer	street character varying(1000)	house_number character varying(256)	zip_code character varying(256)	city character varying(256)	state character varying(256)	country character
56	Vrioulon	16-18	161 21	Kaisariani	Attiki	Greece
92	Smirnis	21	161 21	Kaisariani	Attiki	Greece
3	Vrioulon	14	161 21	Kaisariani	Attiki	Greece

Figure 13. Example #3 of Spatial Query on PostgreSQL Database

3. RESULTS

As mentioned, the final product of this study consists of a CityGML file and KML/COLLADA files. The CityGML file is enriched with information regarding the buildings by the relational database (see Figure 14).

```xml
<bldg:class>3</bldg:class>
<bldg:usage>katoikia</bldg:usage>
<bldg:roofType>plakoskepis</bldg:roofType>
<bldg:measuredHeight uom="urn:ogc:def:uom:UCUM::m">12.35</bldg:measuredHeight>
<bldg:storeysAboveGround>3</bldg:storeysAboveGround>
<bldg:lod2MultiSurface>
  <gml:MultiSurface gml:id="UUID_c86c8e24-e3c5-440e-a646-d012576341ff">
    <gml:surfaceMember>
      <gml:Polygon gml:id="UUID_3c10c6fe-18d6-4de5-8553-0af2899f4310">
        <gml:exterior>
          <gml:LinearRing gml:id="UUID_3c10c6fe-18d6-4de5-8553-0af2899f4310_0_">
            <gml:posList srsDimension="3">23.75620642651575 37.96958438784532 0.0721473960789185
                                          23.756148219637925 37.9695961560728 0.0721473960789183
                                          23.75620642651575 37.96958438784532 0.0801488176295102
                                          23.75620642651575 37.96958438784532 0.0721473960789185</
          </gml:LinearRing>
        </gml:exterior>
      </gml:Polygon>
    </gml:surfaceMember>
  </gml:MultiSurface>
</bldg:lod2MultiSurface>
<bldg:boundedBy>
  <bldg:WallSurface gml:id="UUID_649f29c1-7c6f-4adb-8dc9-74001bed66be">
    <gml:boundedBy>
      <gml:Envelope srsName="urn:ogc:def:crs:EPSG:7.9:4326" srsDimension="3">
        <gml:lowerCorner>23.756147830742766 37.96958576176928 0.0721473960789183</gml:lowerCorner>
        <gml:upperCorner>23.756206815419155 37.96959478214834 12.4253951435158</gml:upperCorner>
      </gml:Envelope>
```

Figure 14. Part of the Exported CityGML File by 3DCityDB

KML/COLLADA files visualize the buildings in 3D applications, like Google Earth in four types: Footprint, Extruded, Geometry and COLLADA (see Figure 15). The exported “Footprint” files visualize the buildings’ footprint on the ground, the “Extruded” files visualize the buildings’ volume and the “Geometry” files visualize the analytical geometry of the buildings’ surfaces and the different colors per type of surface. The “COLLADA” files’ visualization is like the “Geometry” files’ visualization with the extra support of surface textures.

Figure 15. Four Types of KML/COLLADA Files

Furthermore, this implementation supports SQL and thus allows for queries on the database, analysis tasks, schema and data validation as well as spatial data management and mining through the PostGIS extension.

4. CONCLUSIONS

The 3D spatial data are useful in a wide range of applications, especially to those related to urban environment visualization and representation. The Cadastre, the urban planning and commercial activity, disaster/crisis management, energy planning, environmental management, navigation and traffic management as well as virtual tourism are fields that can benefit from 3D technology advancements. The availability of spatial data is increasing steadily as more and more municipalities decide to create virtual 3D city models. Nowadays, it is evident that 3D city models have a positive impact on a large number of governmental and administrative works and procedures leading to better communication, design improvements, lower construction costs, faster and on-time project completion and risk mitigation with a positive overall result for the citizens, the state and the businesses.

CityGML is a model that is already used in a wide range of applications for 3D city modeling. It also enhances the interoperability between many software packages, as it is clear in the current application and represents models of five different LoDs. The CityGML model includes geometrical, topological, semantic and appearance information for the representation of 3D urban objects, leading to a huge amount of 3D city data.

The vast amounts of data require robust spatial databases, such as PostgreSQL, which are able to effectively store and manage large quantities of information and handle overload. Such a relational spatial database includes a comprehensive set of spatial and descriptive information, which allows users to create interactive questions of spatial or descriptive type, analyze spatial data, adapt and export them to analog (e.g. printouts of maps and charts) or digital form (e.g. spatial data files, interactive online maps). Smart and effective management tools for spatial and descriptive data manipulation as well as user-friendly design allow for quick quality solutions to spatial problems, in a comprehensible and easily accessible, by the users, way. In general, information which is necessary for strategic decisions on issues that pertain to a city's operation, can be provided with the use of 3D city models and the data analysis of the relevant database.

The system of the 3D city model and the relational spatial database of this solution finds predominant use in the cadastral applications, since it can deploy the capabilities of a city model's basic structure in 3D Cadastre and generally in 3D Land Administration Systems (LASs). Connection support between the Cadastre and external databases enables the right use of complex 3D cadastral data for each property, their owners and the rights exercised on them.

Another robust example of field use of the current application system is urban planning. This field's main object revolves around the formatting of the city and its public spaces and requires complex procedures that must take into consideration many different spatial components that are involved in the creation of the space, such as the architecture of buildings and landscape, space uses and operations, financial sustainability of all relevant activities, environmental protection and development agenda. A similar example that points to the future is the design of Smart Cities, where the increased level of complexity demands that more detailed models are used as underbeds.

5. ACKNOWLEDGMENTS

This work was supported by the EU FP7 project Four Dimensional Cultural Heritage World (4D CH World) the grant agreement 324523 and the EU H2020 TERPSICHORE project "Transforming Intangible Folkloric Performing Arts into Tangible Choreographic Digital Objects" under the grant agreement 691218.

REFERENCES

El-Mekawy, M., 2010. *Intergrating BIM and GIS for 3D City Modelling – The Case of IFC and CityGML*, Licentiate Thesis, Department of Urban Planning and Environment – Royal Institute of Technology (KTH), Stockholm, Sweden.

Góźdź, K., Pachelski, W., Van Oosterom, P., Coors, V., 2014. *The Possibilities of Using CityGML for 3D Representation of Buildings in the Cadastre*, 4th International Workshop "3D Cadastres", Dubai, United Arab Emirates http://www.gdmc.nl/3DCadastre/literature/3Dcad_2014_34.pdf (9-11 Nov. 2014).

Gröger, G., Kolbe, T. H., Czerwinski, A., Nagel, C., 2008. *OpenGIS® City Geography Markup Language (CityGML) Encoding Standard*, Open Geospatial Consortium Inc., OGC 08-007r1, Version 1.0.0.

International Alliance of Interoperability (IAI), 1999. *IFC Object Model Architecture Guide – Enabling Interoperability in the AEC/FM Industry*, Industry Foundation Classes, Release 2.0, Virginia, United States.

Kolbe, T. H., König, G., Nagel, C., Stadler, A., 2009. *3D-Geo-Database for CityGML*, Documentation, Version 2.0.1, Institute for Geodesy and Geoinformation Science - Technische Universität Berlin, Berlin, Germany.

Kunde, F., Asche, H., Kolbe, T. H., Nagel, C., Herreruela, J., König, G., 2013. *3D City Database for CityGML – 3D City Database Version 2.0.6-postgis, Importer/Exporter Version 1.5.0-postgis*. Tutorial, Release Version, Department of Geography - University of Potsdam, Institute for Geodesy and Geoinformation Science - Technische Universität Berlin, Berlin, Germany.

Stadler, A., Kolbe, T. H., 2007. *Spatio-Semantic Coherence in the Integration of 3D City Models*. 5th International ISPRS Symposium, Commission II, WG II/7, Institute for Geodesy and Geoinformation Science – Technische Universität, Berlin, Germany http://www.isprs.org/proceedings/XXXVI/2-C43/Session1/paper_Stadler.pdf .

Zhu, Q., Li, F., Zhang, Y., 2005. *Unified Representation of Three Dimensional City Models*. ISPRS XXXVI, 4/W6, Workshop "Service and Application of Spatial Data Insfrastructure", Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing - Wuhan University, Hangzhou, China http://www.isprs.org/proceedings/xxxvi/4-w6/papers/237-242qingzhu_a052.pdf (14-16 Oct. 2005).

3DCityDB (The CityGML Database). http://www.3dcitydb.org/3dcitydb/welcome/

CityGML Homepage. http://www.citygml.org/

OGC (Open Geospatial Consortium). http://www.opengeospatial.org/

PostgreSQL Tutorial. http://www.postgresqltutorial.com/

SketchUp. http://www.sketchup.com/