Available online at www.sciencedirect.com

ScienceDirect

Procedia Technology 16 (2014) 310 – 321

Procedia Technology

CENTERIS 2014 - Conference on ENTERprise Information Systems / ProjMAN 2014 - International Conference on Project MANagement / HCIST 2014 - International Conference on Health and Social Care Information Systems and Technologies

Procedural Generation of Traversable Buildings Outlined By Arbitrary Convex Shapes

Telmo Adão [a], Luís Magalhães [b], Emanuel Peres [b], Francisco Pereira [a]

[a]*University of Trás-os-Montes and Alto Douro, Engeneering Department, 5000-801 Vila Real, Portugal*
[b]*INESC TEC (formerly INESC Porto) and UTAD – University of Trás-os-Montes e Alto Douro, 5000-801 Vila Real, Portugal*

Abstract

Virtual reconstructions are commonly used in archaeology to represent cultural heritage monuments that had been lost or damaged by natural causes. Traditionally, these reconstructions require a huge number of human resources and large ranges of time, resulting in high costs of production. To tackle with this issue, many researchers developed semi-automatic techniques to produce virtual models expeditiously. These procedural techniques provide different ways of represent buildings, including interiors and outer facades, in an archaeological or modern context. However, the existing techniques focusing building interiors only support the production of virtual models composed mainly by regular shapes such as rectangles. In this paper it will be presented the first steps of a novel methodology to provide a solution for the generation of building interiors constrained by arbitrary convex shapes. This methodology uses a specific ontology with a set of rules in order to regulate the generation process. The sequence of steps includes the room placement and area definition, section cuts and area readjustments, room linking and finally the extrusions and roof placement to deliver the final 3D model.

Peer-review under responsibility of the Organizing Committee of CENTERIS 2014.

Keywords: Procedural modeling system, Procedural modeling architecture, Arbitrary Geometry, Arbitrary Shape, Virtual Representations, Computer-Aided Engineering, Computer-Aided Design (CAD)

Peer-review under responsibility of the Organizing Committee of CENTERIS 2014.
doi:10.1016/j.protcy.2014.10.097

1. Introduction

Ancient buildings and monuments are commonly used in archeology to study the cultural aspects of past civilizations. However, many of these structures are severely degraded which makes its analysis a very difficult or even impossible task. In not rare cases, this issue leads professionals to use virtual reconstructions as an alternative to real structures. The creative process of producing the required models usually involves several designers working upon the same project, during huge amounts of time. Alternatively, procedural modeling proposes a semi-automatic, cost-effective and fast way of providing accurate virtual representations. During the last two decades, this subarea of computer graphics has explored different methodologies to solve the generation of urban environments and building interiors with interesting results. Moreover, in some archaeological case studies procedural modeling techniques have been used to produce buildings virtual models, revealing the flexibility and extensibility of this researching area. Although, there are one issue that remains unexplored that is the production of virtual models to describe geometrically building interiors with arbitrary shapes. Until now, the existing solutions focused the generation of squared or regular shapes. This kind of shapes are not suffice to support the geometrical description of every type of buildings, leading, sometimes, to hybrid but also laborious solutions based on procedural modeling and traditional approaches using manual techniques. On the other hand, virtual representations of monuments require, in not rare cases, support for building reconstruction considering an arbitrary outline of restriction different from regular squares. To provide the referred requirements and overcome the identified current approaches' limitations, in this paper it is presented a system that constitutes an extension to previous work [1] with a stronger focus in the method for generating geometry provided by the procedural modeling module. This module implements a nouvelle methodology capable of generate and operate with a wider range of geometries, namely convex polygons, aiming the semi-automatic production of buildings constrained by non-regular shapes. The first input of the system is an ontology-based XML (extended markup language) for building definition which is read and verified by an XML moderator module. This XML proposes a standard ontology-based format for building definition designated by XML4BD (extended markup language for building definition), wittingly created for this project. Differently from [1] which uses an XML rigidly ceded by an extraction module susceptible of errors injected by natural language, the proposed format intends to provide a well defined way of configuring a building for more precise and independent operations. Also, a new feature was added to the XML moderator module: the capability of pre-testing geometry in order to balance the occupation of rooms in the available area and also to ensure coherency in models (for example, rooms connected by definition will lose the transition if the pre-test doesn't find an adjacent wall large enough). Module operation results in a populated class structure (ontology-based) which is used by a procedural modeler module implementing the referred methodology to produce the final form of 3D building. Then, this methodology iterates through a set of steps to achieve the expected model: constraint polygon determination, room area division considering room classification (a new feature for room type differentiation, e.g., kitchen or living room), placement of windows and exterior access doors, placement of transitions between rooms, roof generation, placement of ceilings and floors and finally, model detailing. More details about the referred system and methodology can be found throughout this paper. Thus, the document is organized as following: section 2 presents a brief background about procedural modeling of buildings; section 3 exposes the system overview; the adopted ontology, related xml and xml moderator are addressed by section 4 and 5. Section 6 dissects the proposed methodology; section 7 exposes the tests made to the tool, focusing the capabilities of the procedural modeler operating with the proposed XML4BD - used to define a set of fictitious ancient buildings - and, finally, section 8 ends with the final remarks.

2. Background

Several research works have already addressed the semi-automatic generation and reconstruction of buildings in virtual environments. These works can be divided in two classes: the generation of outer façades and the generation of building interiors.

2.1. Generation of Outer Facades

The generation of outer facades was one of the first concerns for procedural modeling researchers. Parish and Muller [2] adapted an L-System [3] to generate Manhattan virtual city. A rewritable alphabet was created in order to generate the roadmap and the same type of generation system was applied to generate buildings in a following phase. Afterwards, Wonka *et al.* [4] created a tridimensional shape grammar to operate directly on the geometries instead of alphabets. The process of split and replace the geometry was used by them to detail building facades. Another remarkable methodology related with outer facades generation is CGA shape [5], where a mass model, composed by volumetric shapes, constitutes the starting step of the process. Afterwards, the outer façades are generated, which gives the exterior appearance of the building. Finally, the doors and windows are placed. The whole process ensures a high level of coherency, avoiding unnatural occurrences such as truncated windows. Coelho *et al.* [6] developed a procedure modeler based on a geospatial L-System in order to produce urban environments. The streets information and blocks location is picked from a database with real coordinates which is then mapped and projected on the virtual world coordinates. The archaeological area is also a concern addressed by a few works. Dylla *et al.* [7] a developed a system to reconstruct Rome virtually with a hybrid solution that partially loads models, created using traditional approaches, and applies them procedural techniques to generate structures. The class of each element defines what kind of approach is needed. For known positions, dimensions and design, class I elements are loaded from pre-created models in a commercial computer aided design software. In case of lack of information, class II elements are generated procedurally using CGA shape methodology. The same methodology was also applied in [8] where Muller *et al.* reconstructed Puuc-style buildings similar to the ones found in Xkipché, México. Another tool, based on ontology, was developed by Liu *et al.* [9] to recover virtually cultural heritage environments, using ancient China as a case study. The city generator uses an ontology and users input to produce the virtual model. The coherence in generation is ensured by a style checker in order to avoid generation errors, such as buildings upon streets. The user input is provided through a grammar containing the buildings definitions.

2.2. Generation Of Building Interiors

The traversable structures are also an issue that has been addressed by a considerable numbers of working methodologies. Chaplin *et al.* [10] developed a shape grammar to generate floor plans with basic rooms and simple geometry. These rooms are categorized in functional zones, according to its function, allowing the coherent placement of furniture. Another approach based on Monte-Carlo algorithm can be found in [11]. The process starts by generating a graph of connected rooms, based on user rules. Then, using the referred Monte-Carlo approach, rooms are placed and expanded until they reach a state of balance and coherence. The lazy generation of building interiors [12] is another proposal focusing real-time generation. In this work the authors suggest a space division method into rectangular rooms and hall passages. At the beginning, a temporary region is defined and then divided into other temporary and construction zones. Division process occurring inside a region is considered finalized when there are no more temporary rooms remaining. The squarified treemap is an interiors generation strategy proposed by Marson and Musse [13]. The method starts by dividing the building square in three distinct functional areas: social, private and service. Then, the squarified treemap is reapplied to subdivide the referred zones in rooms, accordingly with user requirements. Rooms are connected taking into account the connection rules imposed by authors. The final step is the placement of the corridor to connect the unreachable rooms. Mirahmadi and Shami [14] used the same method for splitting rectangular areas with some optimizations at the corridor placement step in order to increase the realism of the architectural designs. Adão et al. [15], on the other hand, used a pure treemap process to generate the floor plan of typical ancient roman houses instead of a squarified approach. The decision of using this approach resulted from a special requirement: the order of the rooms need to be conserved in order to respect architectonic specifications. Differently from the referred works that used squarified treemaps, the corridor is considered since the beginning of the division process. Thus, there is no need to force a corridor placement, which also avoids deforming the planned design. Further steps include the placement of transitions, wall extrusion and roof placement. Tutenel *et al.* [16] used a semantic schema to regulate the generation of building interiors by establishing a relationship between room types. The schema makes easier the production of rules that determine, for example, the room relative position inside its building. The semantic schema also regulates the placement of furniture inside the

rooms, taking in account the functionality of each one (bedroom, kitchen, living room, etc.). Another approach for residential buildings focusing the generation of floor plans is presented in [17], where the methodology receives an input of high level requirements such as building area and number of rooms. Then, a Bayesian network trained with real data is queried in order to obtain data for the floor plan generation. Finally, the walls are extruded from the floor plan to generate a 3D model containing windows, doors and roof. Lopes *et al.* [18] developed a methodology known as constrained growth method to generate floor plans. The rooms are placed upon a grid representing the allotment of the building. The placement takes in account the adjacencies and connections between rooms and the functionalities of each one. The expansion is a final step to adjust rooms inside the grid. The generation of ancient traversable buildings was a concern of Rodrigues *et al.* [19]. The focus of the referred work is the reconstruction of roman houses. An L-System generates the room graph. The methodology also uses the rules of Vitruvius adapted by Maciel [20], to regulate the generation of the structures.

3. System Overview

The main goal of the presented system is the development of an expeditious tool to produce or reconstruct virtual models of ancient monuments in a semi-automatic way. The concept supporting the tool is presented as follows. The system contains a set of prefabricated XML files defining the buildings. Buildings' definition consists in the analysis of ancient floor plans and descriptions which are converted into this XML files based on a normalized building ontology. This XML files passes through an XML moderator module that checks its syntax, loads the informations in a class structure mapping the ontology and finally filters inconsistencies at the floor plan level such as connecting rooms that share a transition without *any* adjacent wall – coherency insurance. After validation, a class structure mapping the ontology is loaded with the XML the procedural modeler generates the archeological building, taking into account a set of additional rules defining architectonic constraints (e.g. wall thickness). The architecture of the developed system is shown on Figure 1.

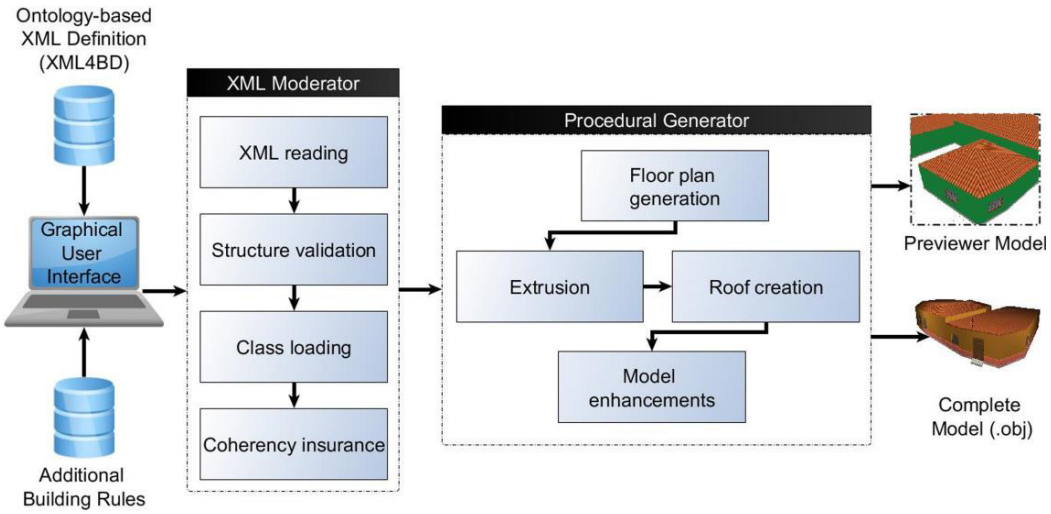


Figure 1. System's architecture: contains a XML Moderator to validate XML input files; also contains a procedural modeling module to generate the geometry of the building.

4. Ontology-based XML and Additional Rules

The system considers XML data files as input. These XMLs must follow a structure based on building ontology: the XML4BD (extended markup language for building definition) structure. Thus, system expects to find the definition of a building, from its outline to the interior rooms. This information will be used to load system class

structure mapping ontology, after XML validation. Taking in account the recently loaded classes, the procedural modeler module iterates through a set of methodological steps to produce the virtual model.

4.1. Ontology

The semantic scheme defined by the ontology - based on [21] - constitutes the first regulatory structure for the present system, creating the awareness of a generic building composition: buildings parts per building, floors per building parts, rooms per floor and related transitions such as doors and windows (Figure 2). This structure provides the possibility of defining a regulated sequence of operations and also the geometric transformations according to each element, maintaining the coherence of the virtual model. Also, a data model was extended from the referred ontology aiming to complete it by pointing out a set of high level requirements or detailing the required structure model. The high level requirements are mandatory informations such as the points defining the polygon that describes the building limits, the number of building parts revealing the horizontal compositions, the number of floors per building part, the number of rooms constituting each floor and the connections between rooms. That kind of information is suffice to generate a virtual model of a building.

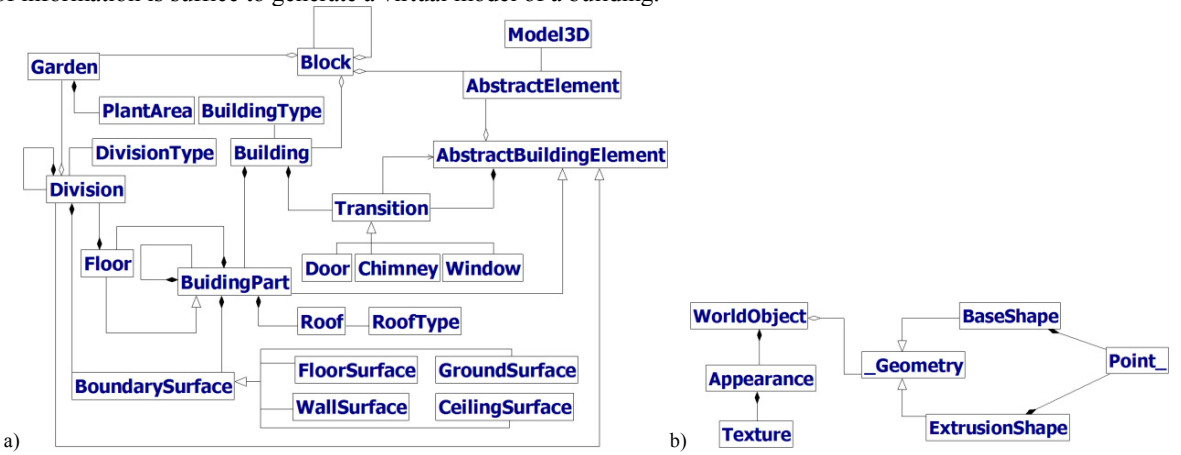


Figure 2. Ontology representation. a) Exposes the ontology that defines the relations between elements that form a building. b) Every object inherits from the class WorldObject which contains a Geometric structure and Appearance, which is used for visualization proposes.

4.2. XML4BD – An extended markup language for building definition

The structure of the adopted XML format, baptized of XML4BD, follows the referred ontology structure. Thus, inside the building node, it is defined the constraint polygon, and building parts – horizontal compositions of the building. Inside each building part node is defined the set of floors and inside each floor node, there's a group of rooms. Inside the building node also specified the connections between rooms.

4.2.1. Building

Building node is the higher one on the hierarchy of nodes. This node declares the following data:

- Set of points - defines a constraint polygon. The constraint polygon is a parameter expecting a set of points;
- Adjacencies - defines a set of connections between inner rooms;
- Building Parts – groups the horizontal structures that compound the building

4.2.2. BuildingPart

A building part represents the horizontal part of the building. It can be seen as a sub container, holding floors. This node declares the following elements or child nodes:

- Roof - defines the type of covering roof. It can currently assume the following values: Flat, Pyramide, Hip, Mansard, MansardHoled and None.
- Height – defines the height of the structure.
- Floors – A list of the floor grouped by this structure
- Building Parts – Optionally, a building part can hold sub building parts for organizational purposes.

4.2.3. Floor

Define a container for rooms, and also some features concerning with the exterior aspect of the building. The child nodes included by Floor are:

- Base height: defines the relative height of the structure that sustains the floor
- Exterior wall height: defines the height of a secondary outer wall.
- Divisions: contains the set of rooms that constitute a floor.

4.2.4. Division

Division is the most atomic structure inside a building. This structure consists in the interior portion of the floor. It's normally connected to other structures of the same kind which allows the mobility inside the building. Its child nodes include:

- Size: Defines a relative occupation of a room. It can be small, medium or big. Further topics will explain the relevance of the value provided by this node for rooms' occupation, using a greedy approach.
- Type: the rooms might be generic rooms, bedrooms, living rooms, corridors, kitchens or toilets.
- IsEntry: flags the room as accessible from the exterior or not.
- Windows: Set of structures that define the windows that a room may contain, at maximum (optional).
- Special structures can be also defined, optionally inside each room node. Currently, the system recognizes gardens and pools.
- Divisions – Optionally, a Room can hold sub rooms for organizational purposes.

4.2.5. Transtions

Transitions are child nodes of building that globally define the connections between rooms, through the definition of adjacencies:

- Adjacency*:* Establishes a connection between two rooms.

5. XML moderator

XML moderator module constitutes the first barrier of regulation before a building virtual model generation. It is responsible for reading each XML4BD file and also for validating its structure against ontology. After structure validation, the module loads a class set which maps ontology accordingly with the XML4DB data. In this step, it is also applied a greedy algorithm to spread the weights of the rooms, according with the size of occupation, until the sum of their weights result in the value 1 – full occupation. Then, a coherency checker tests the geometric model of the floor plan, implicitly loaded on the class structure. The purpose of these tests is to check if the weights and the arrangements of rooms are properly set up. This checking class is responsible to verify if all connecting rooms own a common shared wall with the available space to place door. Coherency checker can also increase and decrease the room weight dynamically in order to try to satisfy the referred condition. If it fails, transitions which present problems are suppressed in order to avoid further geometrical issues in the Procedural Modeling module.

6. Procedural Modeler – room generation constrained by arbitrary shapes

The aim of the procedural modeler is the semi-automatic generator of building virtual models. Its work is very similar to an automatic designer trying to reconstruct information of a certain monument following the orientations of an archaeologist, which is our populated ontology. Thus, the modeler uses the class structure previously loaded and verified by the XML Moderator module in order to obtain building definition. Among the classes, there's a set of mandatory data which includes the polygon that defines the limits of the building, their constituting rooms and

the organization of them in space, the weights of occupation of each one in the restriction polygon and the connections between them. Once the floor plan definition is determined, the modeler starts a progressive process to produce a virtual model, which is represented in Figure 3.

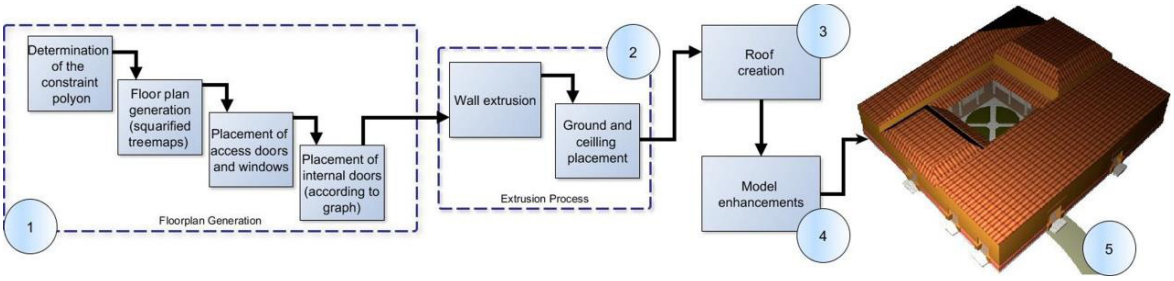


Figure 3. Building generation process in 5 main steps, from the floor plan layout to the complete 3D model. The method expects a constraint polygon (step1.1) that is subdivided into several smaller areas forming the floor plan areas based on treemaps approach (step 1.2). Then, marks for windows and doors are properly placed in the limits of the building to flag connections with the exterior (step1.3). Inner marks are also placed to flag the doors that connect a pair of rooms (step 1.4), according to the room graph provided by ontology-grammar rules. After that, the walls are extruded (step 2.1) taking in account the transitions previously flagged and the roofs and grounds are generated(step 2.2). Finally, the roof is placed (step 3) and some enhancements are made to improve buildings visualization (footers, door frames, windows frame, etc., during step 4). The final result is a full virtual model of the building that can be directly previewed in GUI and also a wavefront file (.obj) that can be imported by the most of the existing CAD tools (e.g. Blender).

The first step concerns with the floor plan generation groups four operations: constraint polygon determination (must be convex, otherwise the modeler forces this requirement using a convex hull algorithm), floor plan area division (according with the loaded class structure, based on XML4BD), placement of external doors and windows and also placement of connecting doors between rooms (correspondently with the graph defined in class structure, loaded from XML4BD input file). The wall extrusion step will raise the walls in the building, taking in account its transitions. The last steps will produce grounds, ceilings and roof and finally the 3D model of the building. The next subsections will describe this methodology in greater detail.

6.1. Restriction Polygon

The polygon that limits the building was previously loaded in building class by the XML4BD input file and will be used to constrain the rooms inside. Specifically, it is a list of points in a XYZ format describing the convex polygon that represents the building frontier. This is also the first input of the modeler.

6.2. Floorplan Generation using a treemaps approach

The floor plan is achieved in two stages: in first place, the modeler splits the convex area to produce the layout of the rooms and then, these rooms are connected. Both stages operate accordingly with the class structure loaded from the XML4BD by the XML Moderator module. The class structure provides the informations of how to split the area, specifying a hierarchical organization for subdivision and the respective weights of occupation. This kind of format promotes the use of a treemaps approach, properly modified to deal with the irregularity of the constraint polygon. Therefore, the modeler iterates through building parts and also rooms, present in the class structure, in order to fulfill a tree structure defining the hierarchic set of containers, sub containers and final rooms. Each container in tree has a weight that results from the cumulative weights defined for its child structures as intend to show (1):

$$container.weight = \sum_{i=0}^{n} room[i].weight \tag{1}$$

During the crossover of the classes related with building parts and rooms, the method splits the polygons recursively until the final rooms, following a treemap approach. Here is an example. Let's consider a given room A in the treemap that is waiting to be partitioned in its child rooms. Let the orientation of the process be vertical and let A be described by a convex geometry. For each child room in A, it is created a squared area with the defined weight (relatively to building area). This square area is set to full horizontal occupation and partial vertical occupation based on its weight – which is geometrically truncated to fit the container. After that, the area of the resulting polygon is measured using Heron's formula (2). This operation is repeated with small increments in the child node square (vertical increments of 0.1 units following Microsoft XNA metrics in this case) until the value of the node area reaches the proper weight in container. The process is then reapplied for every child rooms in this vertical container but the orientation changes to horizontal. And the child nodes belonging to this child nodes follow the same process again with the vertical orientation and so on, recursive and iteratively.

$$A\tau = \sqrt{(\sigma(\sigma-\alpha)(\sigma-\beta)(\sigma-\chi)}\text{, where } \alpha, \beta, \chi \text{ represent the lateral sizes of each triangle and } \sigma \text{ the semi-perimeter given by } \sigma = (\alpha+\beta+\chi)/2 \quad (2)$$

The process to produce the floor plan layout is exemplified in Figure 4, exposing a treemap definition and the respective process of area splitting occurring upon the building constraint polygon.

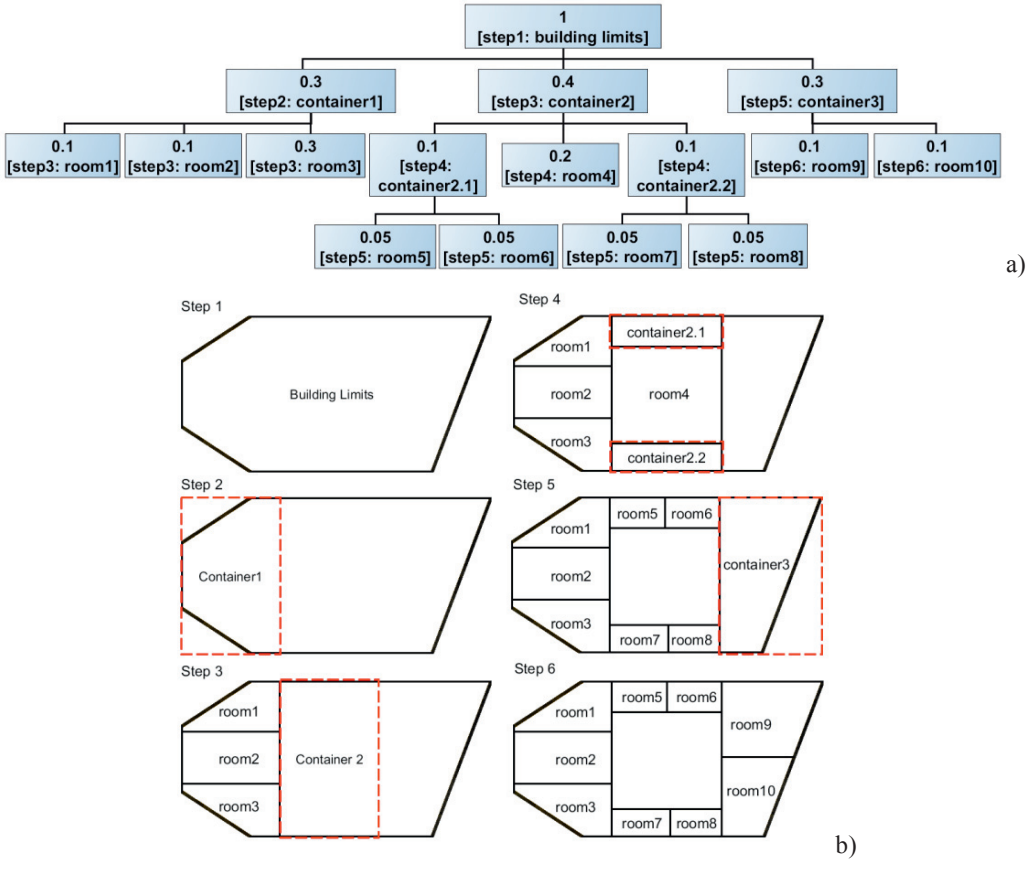


Figure 4. Example of a division process using a floor plan definition. In a) there's a treemap definition, following the hierarchy of rooms loaded to the class structure. b) depicts a set of steps representing the splitting process based on treemap. The first step is to find the total container which matches the outer limits. The second step places the container 1 and iteratively finds and adjusts the polygon that results from the intersection with outer bounds until the required weight of occupation is reached. In step three, the same process occurs inside container 1, but this time to place and expand final rooms. In the remaining steps is applied the same process until the final floor plan is obtained in step 6.

Finally, the connections between rooms are applied based on the class structure that provides informations about adjacencies. These adjacencies establish a set of connections, each one containing a pair of rooms that will allow the creation of a room graph. It will be necessary for placing doors and windows in the later steps.

6.3. Placement of windows and entry doors

After planning the building layout, the modeler marks entry doors and windows in the common wall boundary between the building limits and room limits. For each room, it is verified if the entrance door exists – this is simply a flag provided by a room class object. If so, the door is placed considering also the existence of windows. The common shared wall between the current room and the building outline is determined and then the door is marked providing space for the remaining transitions, which are the windows. Then, the windows are also placed in the common shared wall, taking in account the available space and also avoiding windows overlap. If the shared wall is too small to place the planned number of windows, the modeler suppresses them one by one until the remaining set of windows fit in the referred wall.

6.4. Creating the transitions between rooms

This step strongly depends on the generated graph referred in the first step. Each adjacency in graph contains the pair of rooms identified by a unique id that define a connection. Therefore, it is possible to know what rooms share a transition. Both rooms are localized in the floor plan and then the common shared wall is determined. Finally, the transition door is marked to inform the existence of a door in the following steps.

6.5. Extrusions Process, Roof Creation and model enhancement

The wall extrusion and the placement of ceiling, ground and roof provides the 3D final form. The wall extrusion takes in account the positions occupied by doors and windows. Thus, the extrusion only happens around the defined transitions. Then, accordingly to the geometric limits of each room, the ground and ceiling are placed. The roof is also generated and placed to cover the building limits. Finally, some features – such as doors and windows frames or building base – are added to the model in order to increase the virtual model realism.

7. System trials

To test the capabilities of the procedural modeller it were populated some XML4BD files in order to simulate fictitious ancient structures. These buildings were projected to be outlined by an arbitrary polygon. Each building has different configurations, with different room arrangements, connections and structures. Thus, the fictitious buildings intend to expose the potentialities of the tool in generating considerable varieties of structures, expeditiously. Following the process described in previous sections, the modeller uses a XML4BD file as input. This file is validated by XML moderator, which results in an ontology-based class structure properly loaded with data provided by the input file. The classes are then processed by procedural generator module, which produces the 3D building. Figure 5 provides an example focusing the input XML4BD and the resulting model.

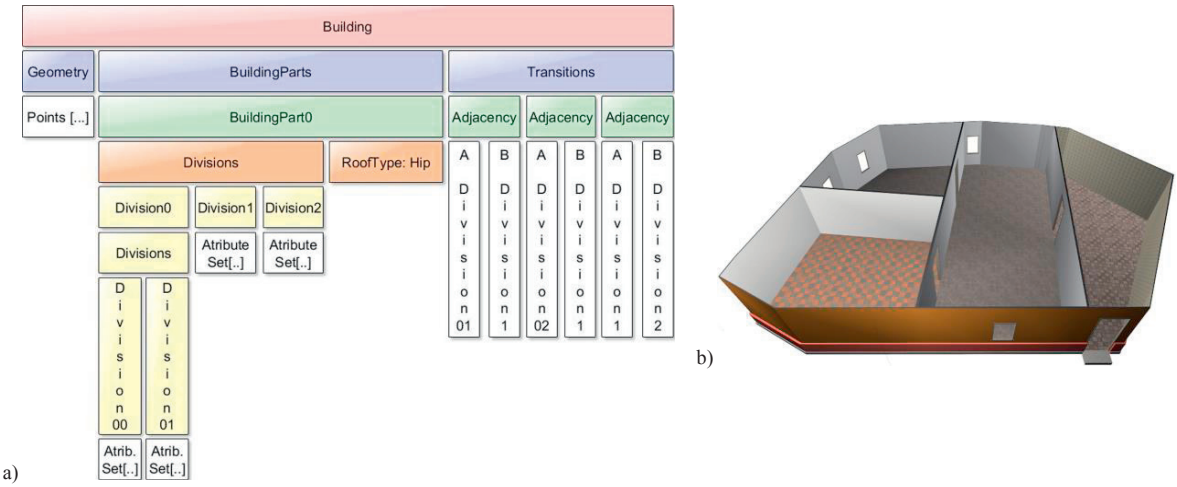
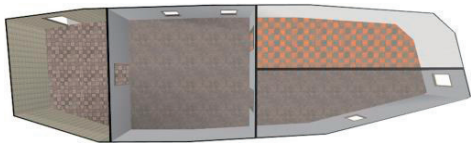
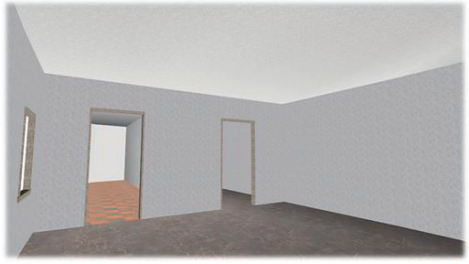


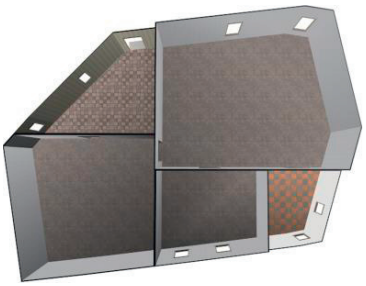
Figure 5. Building model produced from XML4BD: a) graphical view of a XML4BD example, where each block represents a node; b) The building produced accordingly with the presented XML4DB, after being processed by XML Moderator an Procedural Modeling modules.

Three buildings were defined accordingly with the XML4BD format. All of them are constrained by different convex polygon with variable dimensions. The first building is the simplest one. It includes a kitchen – the access room of the building - which is connected to a hall that conducts to a bedroom and also a toilet. Then, a second building is presented with an entry kitchen connected to a living room which also contains a passage to a wider generic room. Generic room interfaces with a bedroom and also a bathroom. Finally, the last building includes 5 rooms, a kitchen and a bathroom. At the center, there's a room that holds a special structure: a garden with four plant areas and also columns. The garden connects all other rooms. Table 1 shows the graphical results of the XML4BDs processing. The examples intend to show the feasibility of the tool in producing different typologies of buildings, constrained by arbitrary convex polygon and with differentiable types of rooms that can also include special structures such as gardens or pools.

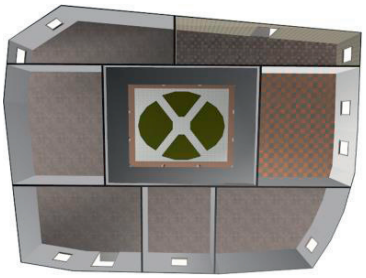
Table 1. Graphical results produced by the system, according with the definitions of three different XML4BD files.

No.	Top view	Interior view
1		

2



3



8. Final Remarks

This paper presented a procedural modeling system that produces 3D traversable buildings constrained by arbitrary convex polygon. The system receives as input an ontology-based XML – also known as XML4BD – which contains the definitions of a certain building, including its related building parts, floors, rooms and transitions. The file passes through an XML moderator that reads and validates XML4BD structure. The referred moderator also loads an ontology-based class structure and performs a few tests to ensure the coherency related with the floor plan (e.g. elimination of transitions between rooms that doesn't share a common wall). Then, the action of the procedural modeling module takes place. This module iterates through the class structure to produce 3D models, from the floor plan step until the last model enhancements.

The trials to the system demonstrated the versatility of this tool in creating buildings with several configurations and topologies. Trial results proved that this tool is effective in producing buildings constrained by pure convex shapes. Besides, it is also possible to include in the building special rooms holding inner structures like pools or gardens with columns.

Acknowledgements

This work is partially supported by the European Fund for Regional Progress - FEDER (Fundo Europeu de Desenvolvimento Regional) through the project 2014/038803 entitled "MixAR – Adaptive Mixed System for Archaeological Sites".

References

[1] Adão T, Batista R, Peres E, Magalhães, L G M, Coelho A. Reconstructing traversable buildings for archaeology with ERAS. In: Proceedings of the Virtual and Networked Organizations: Emergent Technologies and Tools – ViNOrg '13; 2013.

[2] Parish Y I H, Müller P. Procedural modeling of cities. In: Proceedings of the 28th annual conference on Computer graphics and interactive techniques; 2001.

[3] Lindenmayer A, Mathematical models for cellular interactions in development II. Simple and branching filaments with two-sided inputs, Journal of Theoretical Biology 1968; 18:300-315.

[4] Wonka P, Wimmer M, Sillion F, Ribarsky W. Instant architecture. In: ACM SIGGRAPH 2003 Papers, San Diego, California; 2003.

[5] Müller P, Wonka P, Haegler S, Ulmer A, Gool L. Procedural Modeling of Buildings. ACM Transactions on Graphics (TOG) 2006; 25(3):614-623.

[6] Coelho A, Bessa M, Sousa A, Ferreira F. Expeditious Modeling Of Virtual Urban Environments With Geospatial L-Systems, Computer Graphics Forum 2007; 26(4):769-782.

[7] Dylla K, Müller P, Ulmer A, Haegler S, Frischer B. Rome Reborn 2.0: A Case Study of Virtual City Reconstruction Using Procedural Modeling Techniques. In: The Making History Interactive. 37th Proceedings of the CAA Conference, Williamsburg, Virginia; 2009.

[8] Müller P, Vereenooghe T, Wonka P, Paap I, Gool L. Procedural 3D Reconstruction of Puuc Buildings in Xkipche. In: Proceedings of Eurographics Symposium on Virtual Reality, Archaeology and Cultural Heritage (VAST 06); 2006. p. 139–146.

[9] Liu Y, Xu C, Zhang Q, Pan Y. The Smart Architect: Scalable Ontology-Based Modeling of Ancient Chinese Architectures, Intelligent Systems 2008, IEEE, 23:49-56.

[10] Rau-Chaplin A, Mackay-Lyons B, Spierenburg P. The Lahave house project: Towards an automated architectural design service. In: Proc. Int. Conf. Comput.-Aided Design, Hagenberg, Austria; 1996. p. 25–31.

[11] Martin J. Procedural House Generation: A method for dynamically generating floor plans. In: Symposium on Interactive 3D Graphics and Games; 2006.

[12] Hahn E, Bose P, Whitehead A. Lazy Generation of Building Interiors in Realtime. In: Electrical and Computer Engineering '06 Canadian Conference on (CCECE); 2006. p. 2441-2444.

[13] Marson F, Musse S R. Automatic Real-Time Generation of Floor Plans Based on Squarified Treemaps Algorithm, International Journal of Computer Games Technology 2010, vol. 2010.

[14] Maysam M, Shami A. "A novel algorithm for real-time procedural generation of building floor plans," arXiv preprint arXiv:1211.5842, 2012.

[15] Adão T, Magalhães L, Peres E. Semi-automatic Virtual Reconstruction of Ancient Roman Houses. In: CISTI'2013 - 8th Iberian Conference on Information Systems and Technologies; 2013. p. 969-974. Lisboa, Portugal.

[16] Tutenel T, Bidarra R, Smelik R. Rule-based layout solving and its application to procedural interior generation. In: CASA Workshop 3D Adv. Media Gaming Simul; 2009. p. 15-24. Amsterdam, The Netherlands.

[17] Schkufza E, Merrell P, Koltun V. Computer-generated residential building layouts, ACM Transactions on Graphics (TOG) 2010; 29:1-12.

[18] Lopes R, Tutenel T, Smelik R M, Kraker K J, Bidarra R. A constrained growth method for procedural floor plan generation. In: Proc. 11th Int. Conf. Intell. Games Simul; 2010.

[19] Rodrigues N, Dionísio M, Gonçalves A, Magalhães L M G, Moura J P, Chalmers A. Rule-based Generation of Houses. Computer Graphics & Geometry 2008; 10(2): 49 – 65.

[20] Maciel M. Vitrúvio - Tratado De Arquitetura. Ist Press; 2006.

[21] Adão T, Magalhães L M G, Bessa M, Coelho A, Sousa A, Rodrigues N, Rodrigues R, Pereira F, Moura J P, Reis L. ERAS – An Ontology-Based Tool for Expeditious Reconstruction of Virtual Cultural Heritage Sites. In: 20º Encontro Português de Computação Gráfica; 2012. Viana do Castelo, Portugal.