Topologic: Exploring spatial reasoning through geometry, topology, and semantics

Wassim Jabi[1] and Aikaterini Chatzivasileiadi[1]

[1] Cardiff University, Cardiff CF10 3NB, UK
{jabiw, chatzivasileiadia}@cardiff.ac.uk

Abstract. This paper presents Topologic, a software modelling library that supports a comprehensive conceptual framework for the hierarchical spatial representation of buildings based on the data structures and concepts of non-manifold topology (NMT). Topologic supports conceptual design and spatial reasoning through the integration of geometry, topology, and semantics. This enables architects and designers to reflect on their design decisions before the complexities of building information modelling (BIM) set in. The paper summarizes related work on NMT starting in the late 1980s, describes Topologic's software architecture, methods, and classes and discusses how Topologic's features support conceptual design and spatial reasoning. The paper includes a report on a software usability workshop that was conducted to validate a software evaluation methodology and reports on the collected qualitative data. The paper concludes with reflection on Topologic's features and how it enables a shift from pursuing fidelity of design form to pursuing fidelity of design intent.

Keywords: Conceptual Design; Geometry; Semantics; Spatial Reasoning; Topology.

1 Introduction

The advent of building information modelling (BIM) systems has revolutionised the architecture, engineering and construction (AEC) sector. By allowing architects, engineers, consultants and contractors to collaborate on a project and share structured and rich information that includes three-dimensional models and federated databases, BIM systems have helped increase coordination and productivity and reduce mistakes [23]. However, while BIM systems have largely served the design development and construction phases well, they have not been as effective in the early phases of conceptual design. This is mainly due to three issues. First, BIM systems focus on creating building components that are highly specified. Designers in the early phases may not know what specific materials, or even forms, they wish to use. They usually employ abstraction as a method to address complexity and postpone specification to a later date. Second, BIM systems, while adept at placing and aggregating components, do not have a formal system of connecting these components. The early phases of design are usually characterised by solving topological issues of adjacency, access, and circulation. Yet, BIM components usually do not have intrinsic adjacency information. Finally, this lack of formal topological information reduces what can be termed the component's spatial awareness – its ability to intelligently react to spatial changes in its surrounding. While BIM components certainly react to user actions and changes in design, they do not always do so correctly and take into account the adjacent spaces because the knowledge of how to react is embedded in the algorithm and not formalized in the data structures. The complexity, lack of formal topology, and resulting difficulties with spatial awareness means that BIM systems struggle to support conceptual design and spatial reasoning. For example, it is usually difficult for BIM systems to conduct shortest path analyses in three-dimensions or report on the centrality or isolation metrics of certain spaces. To solve these issues, a plethora of plugins have been introduced to help with some of the limitations found in BIM software and improve their functionality. Yet these plugins are not generalizable because they must restructure BIM data to satisfy software functionality.

To address the above issues, the authors have been conducting a funded research project that investigates standardised methods for creating connected three-dimensional entities based on non-manifold topology (NMT) [4, 9, 22]. This has resulted in a comprehensive conceptual framework for the hierarchical spatial representation of buildings implemented in a software modelling library named Topologic [19]. While other features of Topologic have been published elsewhere [3, 11, 18, 20], the focus of this paper is on Topologic's support for conceptual design and spatial reasoning by integrating geometry, topology, and semantics.

The remainder of the paper is organised into five sections. Section 2 summarises related work on NMT starting in the late 1980s. Section 3 describes the aims of Topologic, its software architecture, methods, and classes. Section 4 discusses how Topologic's features support spatial reasoning and conceptual thinking. Section 5 reports on a workshop that was conducted internally to test a software evaluation methodology. In this section, we describe three prototypical workflows in Topologic and some of the qualitative data that we collected from the workshop. Finally, section 6 offers concluding remarks on the role of Topologic in the conceptual design phase.

2 Related Work

The realisation that NMT has benefits to offer in building information modelling goes back to the late 1980s and early 1990s. As part of his Ph.D. dissertation at the Rensselaer Polytechnic Institute, Kevin Weiler invented the winged edge data structures and other topological concepts needed for non-manifold topology modelling [33]. In 1989, Masuda et al. created a non-manifold geometric system that can manipulate entities of different dimensionalities in one architecture [24]. In their paper, they provide the mathematical definition of non-manifold geometric models and introduce the Euler operations needed to manipulate them. In 1991, Crocker and Reinke presented non-manifold topology as a new approach "[…] to overcome editing and coverage limitations associated with the manifold nature of existing boundary evaluation schemes" [12]. Their system extended 3D Boolean operations to include surfaces and wireframes in addition to solids. In 1992, Björk analysed building product models that included the topology of spaces and their boundaries [6]. In anticipation of the issues with which BIM systems currently struggle, Björk posits that building models should contain the appropriate data structures to capture the semantics needed to allow design and construction stakeholders to extract the information they need. In 1997, Cavalcanti et al. introduced a new methodology for subdividing the three-dimensional entities from a set of surface patches [9]. Their system maintained an explicit representation of cells forming a non-homogeneous object and enabled its creation using operations that resemble constructive solid geometry. Similarly in 1997, Chang and Woodbury discussed a non-manifold-based representational scheme for efficient exploration of design spaces [10]. They invented a labelling scheme on top of a non-manifold cell-complex to represent entities of different dimensionalities. Their state representation (*srep*) was bi-partite. The first part contained an exhaustive subdivision of space using 0D, 1D, 2D, and 3D entities and the second part contained attribute sets that listed different collections of cells. This scheme was used to build a generative design system (SEED-Config) based on the ACIS non-manifold 3D modeler [31]. In 2009, Lee et al. used a top-down space subdivision method based on non-manifold topology for the modelling of ship compartments [22]. In 2010, Boguslawski and Gold introduced the dual half edge (DHE) structure that allowed them to represent simultaneously the primal and dual graph of non-manifold entities [7]. As we will see below, dual graphs are essential for querying and visualising topological information and in helping with navigation between topological cells. Boguslawski later used this feature as a representation in escape route planning for tall buildings [8]. In 2017, Furiani et al. at Tre University in Rome, Italy, started the development of a system that uses sparse integer arrays for geometric computing of cells of a very general type [14]. Their system, aimed for biomedical imaging, neural networks, and image analysis research was written using the novel Julia Programming

Language. Their project aims to compute the cells of an unknown arrangement and to provide the numerical and topological tools to other algorithms.

3 Topologic

Topologic is intended as a bridge between the conceptual design phase and the BIM-based design development phase. It brings forward building performance analyses including spatial reasoning so that architects and designers can reflect on their decisions and rigorously conceptualize their design before the complexities of BIM set in. Topologic's structured data can then be used in conjunction with intelligent rule-based systems to populate BIM components and to further develop the design within a BIM platform. In this paper, we will focus on Topologic's ability to conduct formal spatial reasoning and semantics-based geometrical and topological operations and illustrate those through detailed workflow descriptions.

Topologic is designed as a core library, written in C++ with additional software layers that expose its methods and data structures to visual data flow programming (VDFP) applications such as Autodesk Dynamo and Grasshopper3D. At the lowest layer, Topologic uses Open CASCADE, an open-source NMT geometry software development kit that provides data structures and modelling algorithms for 3D solid structures [26]. TopologicCore implements the core topologic classes and methods using an object-oriented programming (OOP) approach with additional support utilities. To enable communication with 3rd party software, we implemented an interface layer, written in the .NET C++/CLI language [13].

Topologic's classes include: Vertex, Edge, Wire, Face, Shell, Cell, CellComplex, Cluster, Topology, Graph, Aperture, Content, Context, and Dictionary (see Fig. 1). A vertex is a point in 3D space with X, Y, Z coordinates. A start Vertex and an end Vertex create an Edge. A set of connected edges creates a Wire. A closed set of one or more wires can form the basis of a Face. A set of connected faces creates a Shell. A closed Shell creates a Cell. A set of connected cells creates a CellComplex. A Cluster is a grouping of topologies of any dimensionality. At this point, it is important to draw a distinction between geometry and topology. An Edge can be used to illustrate this distinction. Topologically, an Edge can be described as a border between two regions or a connection between two locations regardless of its geometric complexity. In Topologic, an Edge can represent a geometry of arbitrary complexity and that geometry can be retrieved on demand.

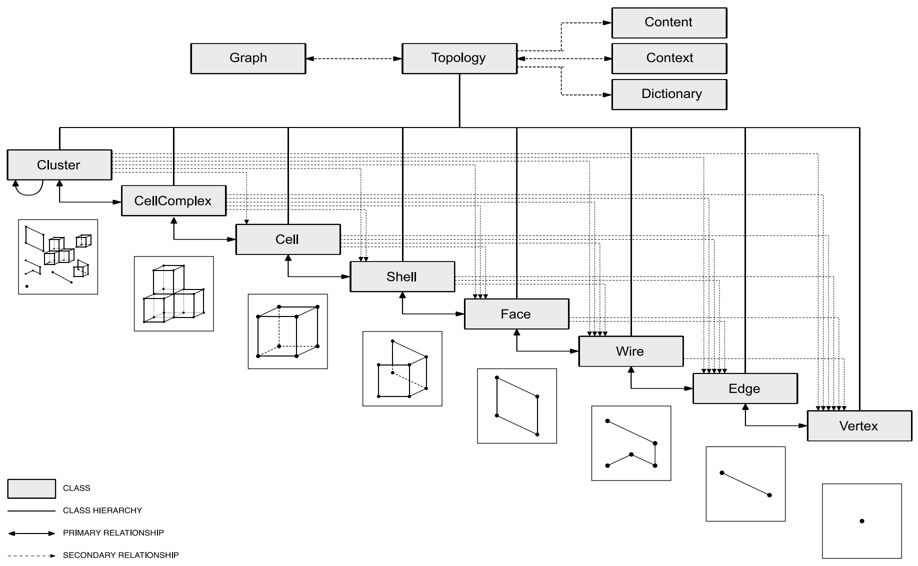

Fig. 1. Topologic's core class hierarchy.

A Graph in Topologic is an extension of the more classical dual graph [8], and is derived from other topologies. The Graph class and associated methods are based on graph theory [32]. A Graph is composed of vertices and edges that connect them. A Graph in Topologic accepts as input any topology with additional optional parameters and outputs a Graph object. In its simplest form, the dual graph of a CellComplex is a Graph that connects the centroids of adjacent Cells with a straight Edge. The topology of a Graph of a CellComplex is a wire (see Fig. 2). As an option, a graph in Topologic can create edges that connect the centres of cells to the centres of apertures and between cells only if they share not only a face, but also an aperture within that face. An aperture is a special type of face that is hosted by another face (e.g. doors and windows). By building a formal dual graph Topologic can conduct full three-dimensional spatial integration/centrality analysis based on graph theory and space syntax theory [16] (see Fig. 3).

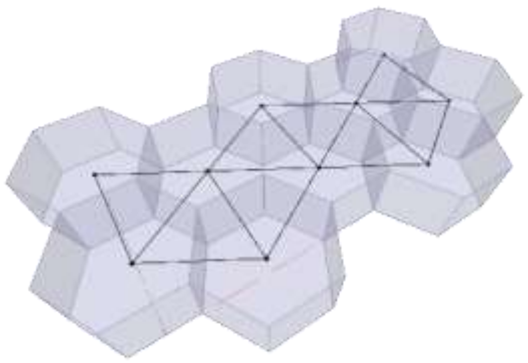

Fig. 2. A primal graph (CellComplex) and its associated dual graph (Wire).

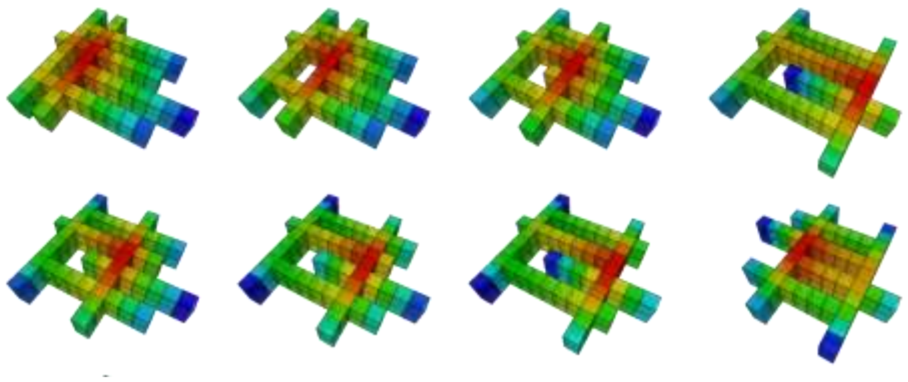

Fig. 3. Graph-based space integration/centrality study conducted in Topologic.

Unrelated topologies can be linked together hierarchically through Topologic's Content/Context system where any topology can store one or more other topologies. Reversely, these content (child) topologies maintain a pointer back to their context (parent) topology. This is useful when one wants to associate topologies that are not linked intrinsically through the boundary representation hierarchy.

Finally, any topology in Topologic can contain a dictionary made of one or more key/value pairs. A key is any identifying string of characters (e.g. "ID") and its value can be of any data type (e.g. a number, a string). When subjected to geometric operations, operand topologies transmit their dictionaries to resulting topologies. For example, if a cell has a dictionary with key/value pair ["ID", "Cell A"] and is sliced into smaller cells, then each of the resulting cells gains a copy of the original dictionary. In addition, if two cells intersect, the intersection aggregates the key/value pairs of its operands. Furthermore, when a dual graph is created, topologies of the primal graph transmit their dictionary to their counterpart topology in the dual graph. This allows for semantic information to be associated with topologies.

4 Spatial Reasoning

The central premise and current trajectory of BIM systems is to focus on creating highly detailed digital models (digital twins) that attempt to accurately represent the three-dimensional form and inventory of the project [27]. This creates a gap in the semantic content of BIM models. This is because BIM models lack a sophisticated ability to capture design intent and logic – the concept of the design. One of the main goals of Topologic is to address this gap by helping architects and designers reflect on, reason about, represent and encode the spatial and conceptual qualities and logic of their design. The aim is to shift design thinking in the early stages from pursuing a fidelity of design form to pursuing a fidelity of design concept. This can be achieved by supporting the designers' need for abstraction, awareness, and conceptual thinking through the tools of geometry, topology, and semantics.

Architectural design, by its nature, is a continuous process of creating geometric and non-geometric artefacts that stand in, represent, and abstract reality [25, 30, 34]. Architectural construction has traditionally been the process of interpreting and realising these abstracted or idealised models [2]. Topologic's classes of objects such as vertices, edges, wires, and faces etc. act as geometric abstractions and placeholders of more detailed representations. For example, a vertex can be the location of an electric outlet, an edge can be a column's centreline, and a face can be the host surface for a curtain wall. Through a custom dictionary and a content/context system, topologies can hold the information, including more detailed geometries, that are required for future design detailing work.

Through topology (i.e. membership, adjacency and connectivity), these entities transcend mere geometry by possessing and utilising intrinsic awareness of their context. In Topologic, the user can query a cell for its faces, edges, or vertices and a unique, non-overlapping list is generated for each category. These returned entities can in turn, be asked to return their constituent members. For example, a cell can return its faces and then ask each returned face for its edges and each returned edge for its vertices. Beyond this top-down query, you can also climb the hierarchy. For example, you can ask a cell to return its vertices, but then you can ask each vertex to return the edges to which it belongs. Similarly, you can ask a cell for its edges and then ask each edge to return the faces that share it.

Conceptual design begins to be supported when you combine these topological queries with other types of semantic queries. As we have seen above, vertices know what edges are connected to them and as such they can query each edge for a certain value in its dictionary or its geometry. Based on the response, the vertex in question can deploy one of several geometric solutions – either by choosing from a list of alternatives in its contents list, or by parametrically modifying its own geometry. One of the interesting features of Topologic is that Boolean operations can manipulate and be manipulated by a topology's semantic information. For example, slicing a cellular building volume with a set of horizontal planes to create floors usually slices all cells/spaces within that building. However, a cell can be assigned a space type (e.g. an atrium) that causes it to resist the slicing and continue to be an undivided volume (see Fig. 4). By simply modifying the semantic assignment, while leaving the remainder of the workflow unchanged, one can achieve a geometrically and topologically different result. Thus, by altering the semantics in the model, the user can experiment with different results and optimise their design. The sum of these abilities enables the designer to think conceptually and strategically about their design. They no longer simply aggregate entities in a display list but connect entities that need to be connected and build conceptual models that react logically and provide meaningful information when queried.

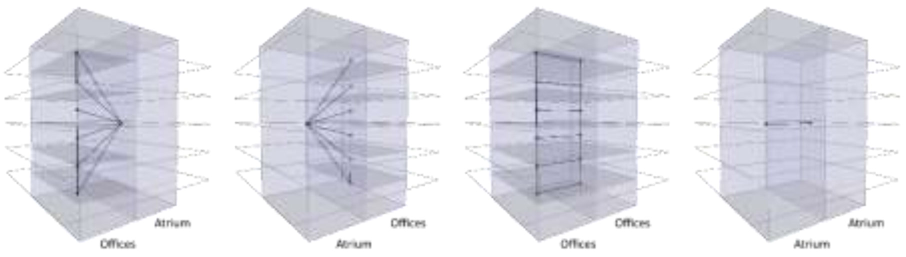

Fig. 4. Semantics-based Boolean operations.

5 Usability Workshop

We conducted a software usability workshop where a handful of participants were asked to complete three workflows and then answer a questionnaire and engage in a discussion with the authors. The aim of the workshop was two-fold. First, we wanted to evaluate and collect initial feedback on the above concepts and workflows, and second, we wanted to test and verify that the basic evaluation methodology is valid in preparation for conducting a formal and a larger scale evaluation on which we will report in future publications.

5.1 The tasks

Three tasks involving creating workflows using Topologic that are relevant to the theoretical concepts described earlier are presented below. In workflow 1 ‘Geometry and Semantics’, the user can explore how information embedded in the dictionaries of topologies can be used to affect the result. Two simple orthogonal intersecting prisms are created. Each prism has a custom dictionary with an identical key (“height”) and a custom value assigned to it. These values are independent of the actual geometric height of the prisms. The goal is to use this information later to customize the height of the resulting intersection. The Boolean intersection of these two prisms (indicated below in red) would inherit the custom dictionary values of the two prisms. For example, if one prism has a “height” value of 2 and the other prism has a “height” value of 5 then the resulting intersection will have a dictionary in which the “height” key has a value of {2, 5} (i.e. a list of the two values of the original dictionaries). The algorithm would then retrieve and add these two values and use them to vertically scale the intersection geometry (see Fig. 5).

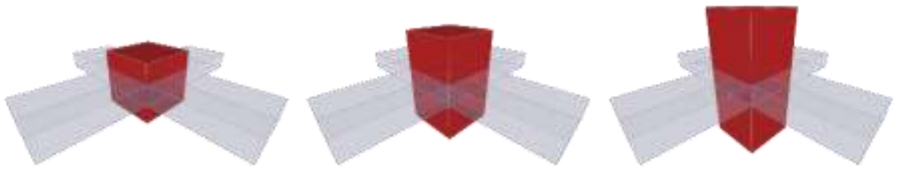

Fig. 5. Two intersecting prisms each with a custom dictionary.

In workflow 2 ‘Shortest path by key’, the user can compute the shortest path between two vertices. The solution takes into consideration the embedded information using graphs and dictionaries. Initially the user creates Cells with varying heights and combines them into a CellComplex. The next step uses the CellComplex to build a graph and compute the shortest path (see Fig. 6). Each Cell in the CellComplex (and therefore each Vertex) in the graph would have a custom dictionary with an identical key (“volume”) and a custom value assigned to it. The goal is to compute the shortest path that goes through the smallest volumes. Input number sliders would be used to change the individual heights of the Cells thus changing their volume. The path would consequently be automatically routed through the Cells with the smallest total volume.

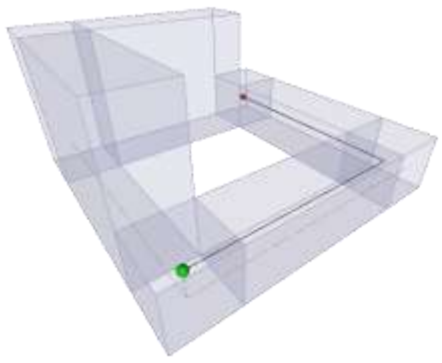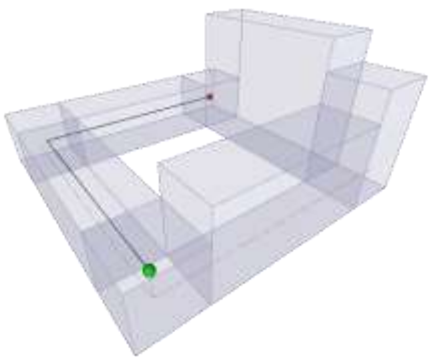

Fig. 6. Shortest path routing through the cells with the smallest overall volume.

Workflow 3 'Topological distances' enables the user to visualise the integration/centrality of spaces in a building [17], by creating a dual graph of the building. The centrality measure of a space is computed by counting the number of Edges in the shortest path possible between the space in question and every other space in the building. The degree of integration or isolation of a space is indicated using size and colour of spheres placed in each space. The larger the size and the warmer the colour of a sphere the more central/integrated it is and vice versa (see Fig. 7).

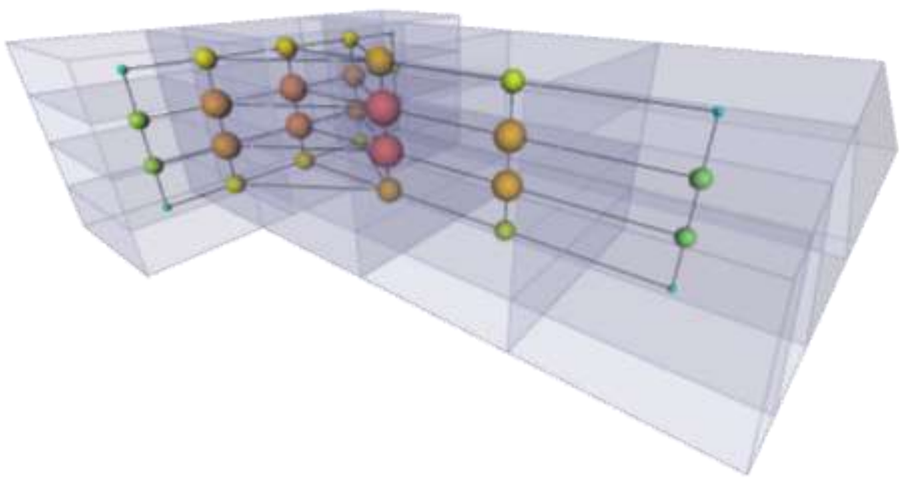

Fig. 7. Visualization of space centrality.

5.2 Evaluation design and early qualitative results

A usability exercise was conducted to gauge initial perceptions of the user experience regarding the three example workflows presented earlier. These are representative of the type of tasks that a Topologic user might do [1]. A handout including the steps required to implement these was shared with the participants. After completing the workflows, they were asked to share their experience through a questionnaire and a discussion with the authors [15, 21, 28, 29]. This qualitative data, while of moderate reliability, served as a preliminary testbed which the authors will use for the setup of a formal usability experiment in the future.

Participants. Eight postgraduate students who had a similar profile to the targeted end users of the tool participated in the workshop [1, 5]. During the exercise, the authors mainly observed the participants completing the tasks with limited communication [5]. Five of the eight participants were at an intermediate level in using VDFP tools, one was an advanced user, while the remaining two were beginners (see Fig. 8). Seven participants were at a beginner level in Dynamo and Topologic, while one was an intermediate user in both.

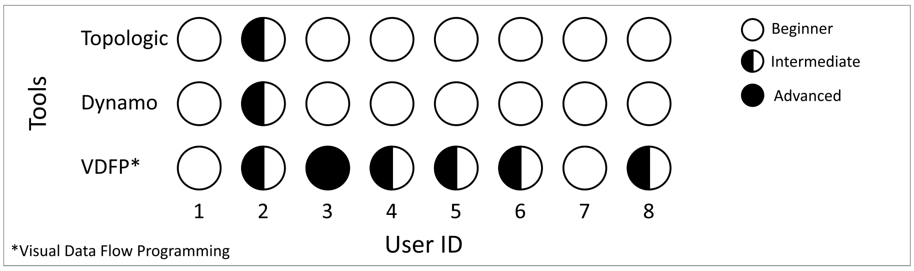

Fig. 8. Level of participants' experience.

Results. All participants were able to complete the given tasks and generally had a positive experience using Topologic. The easy workflows and the fact that scripting can be avoided in some cases were beneficial aspects of Topologic. Some of the collected positive comments were: "pretty straightforward to understand", "this approach of Topologic is an added value and can be pretty useful," and "this function of Topologic would be useful in considering the correlation between different spaces in a building which can help in the design process. This could be used to measure qualitative parameters, such as social interaction […]." On the critical side, users wrote that they "needed to learn the Topologic definitions, e.g. Cells, CellComplex etc., before using the tool in addition to some computer science information, such as the dictionaries concept" and that "without examples to learn the processes and components, [they] would likely find it difficult/timely to [get to know it]. Therefore, links to tutorials/example files would allow users to very quickly pick up Topologic." One user wrote that they are "not sure if [they] would ever undertake such tasks frequently." Other aspects that were confusing were the lack of individual icons for the Topologic nodes and the similar appearance of Dynamo and Topologic nodes, that made it difficult to tell what node belonged to which application. Finally, a limitation of the tool addressed naming conventions and confusion around these, with a typical example being the *Topology.Geometry* and *Topology.ByGeometry* nodes.

Clearly, whether a user would undertake such tasks depends on their interests. It was found that the semantics-based Boolean operations (Task 1) provided the most straightforward approach, while the shortest path analysis (Task 2) required an in-depth study of the theoretical framework prior to engaging with the task. Users seemed to relate more with the spatial integration/centrality analysis through graph theory and space syntax (Task 3), as they were able to link it to practical applications, such as measuring social interactions.

One of the main hurdles of this evaluation was that Topologic is dependent on the host application's interface and naming conventions, so it was difficult to isolate the user experience and focus it solely on the usability of Topologic, instead of the combined usability of Dynamo and Topologic. Moreover, as Topologic is a relatively new tool to be used in conjunction with VDFP applications, most of the participants were at beginner's level with it. The situation was the same regarding the participants' level of experience with Dynamo, yet most of them had an intermediate level of experience with other VDFP applications. Considering the sample of participants, it is anticipated that an intermediate or advanced Dynamo user could potentially give more reliable results in terms of the usability and learnability of the tool. A larger and more diverse sample of participants in terms of experience can yield more reliable results. Additionally, we can give participants the terminology prior to the evaluation so that they familiarize themselves with the concepts. Through the process of conducting this workshop, we had the opportunity to identify parts in the methodology that can be improved regarding the setup of the formal and systematic usability experiment which will allow us to address them in the future and perform a formal and systematic usability experiment with a larger and more diverse group of participants.

6 Conclusion

In this paper, we presented a new perspective on supporting conceptual design and spatial reasoning through the integration of geometry, topology, and semantics. We described the architecture and features of Topologic, a 3D modelling software library that supports the hierarchical representation of space in architecture and reported on the results from a preliminary evaluation study. Fundamentally, Topologic expands the pursuit of fidelity to design form to include the pursuit of fidelity to design intent. This research has indicated that Topologic can use abstraction as a method to address complexity and provide a formal system for creating, editing, and connecting topologies. Components in Topologic have inherent spatial awareness that enables them to intelligently react to requests and changes in their surroundings. Topologic shifts the needed information from ad hoc algorithms to formalized data structures and methods implemented using an object-oriented approach. When combined, these features enable Topologic to support rigorous spatial reasoning and analysis in the conceptual design phase.

References

1. Abdinnour-Helm, S.F., Chaparro, B.S., Farmer, S.M.: Using the end-user computing satisfaction (EUCS) instrument to measure satisfaction with a web site. Decis. Sci. 36 (2), 341–364 (2005).
2. Aish, R.: MicroStation/J. In: Noor, A.K. and Malone, J.B. (eds.) Next Generation CAD/ CAM/CAE Systems, NASA Conference Publication 3357. pp. 167–180. National Aeronautics and Space Administration, Hampton, Virginia (1997).
3. Aish, R., Jabi, W., Lannon, S., Wardhana, N.M., Chatzivasileiadi, A.: Topologic : tools to explore architectural topology. In: Hesselgren, L., Kilian, A., Malek, S., Olsson, K.-G., Olga, S.-H., and Williams, C. (eds.) Advances in Architectural Geometry 2018. pp. 316–341. Klein Publishing GmbH (2018).
4. Aish, R., Pratap, A.: Spatial information modeling of buildings using non-manifold topology with ASM and DesignScript. In: Hesselgren, L., Sharma, S., Wallner, J., Baldassini, N., Bompas, P., and Raynaud, J. (eds.) Advances in Architectural Geometry 2012. pp. 25–36. Springer Vienna, Vienna (2013).
5. Bandi, A., Heeler, P.: Usability testing: a software engineering perspective. In: 2013 International Conference on Human Computer Interactions (ICHCI). pp. 1–8. IEEE (2013).
6. Björk, B.-C.: A conceptual model of spaces, space boundaries and enclosing structures. Autom. Constr. 1 (3), 193–214 (1992).
7. Boguslawski, P., Gold, C.: Euler operators and navigation of multi-shell building models. In: Neutens, T. and Maeyer, P. (eds.) Developments in 3D Geo-Information Sciences. Lecture Notes in Geoinformation and Cartography. pp. 1–16. Springer (2010).
8. Boguslawski, P., Gold, C.: The dual half-edge—a topological primal/dual data structure and construction operators for modelling and manipulating cell complexes. ISPRS Int. J. Geo-Information. 5 (2), 1–20 (2016).
9. Cavalcanti, P.R., Carvalho, P.C.P., Martha, L.F.: Non-manifold modelling: an approach based on spatial subdivision. Comput. Des. 29 (3), 209–220 (1997).
10. Chang, T.W., Woodbury, R.: Efficient design spaces of non-manifold solids. In: Liu, Y., Tsou, J.-Y., and Hou, J. (eds.) Second Conference on Computer-Aided Architectural Design Research in Asia. pp. 335–344. CAADRIA, Taiwan (1997).
11. Chatzivasileiadi, A., Lannon, S., Jabi, W., Wardhana, N.M., Aish, R.: Addressing pathways to energy modelling through non-manifold topology. In: Macumber, D., Meggers, F., and Rockcastle, S. (eds.) SIMAUD 2018: Symposium for Architecture and Urban Design. Society for Modeling & Simulation International (SCS), Delft, the Netherlands, the Netherlands (2018).
12. Crocker, G.A., Reinke, W.F.: An editable nonmanifold boundary representation. IEEE Comput. Graph. Appl. 11 (2), 39–51 (1991).

13. Ecma International: C++/CLI language specification. , Geneva (2005).
14. Furiani, F., Martella, G., Paoluzzi, A.: Geometric Computing with Chain Complexes Design and Features of a Julia Package, (2017).
15. Hassenzahl, M.: The effect of perceived hedonic quality on product appealingness. Int. J. Hum. Comput. Interact. 13 (4), 481–499 (2001).
16. Hillier, B., Hanson, J.: The social logic of space. Cambridge University Press (1984).
17. Hillier, B., Leaman, A., Stansall, P., Bedford, M.: Space syntax. Environ. Plan. B Plan. Des. 3 147–185 (1976).
18. Jabi, W.: Linking design and simulation using non-manifold topology. Archit. Sci. Rev. 59 (4), 323–334 (2016).
19. Jabi, W., Aish, R., Lannon, S., Chatzivasileiadi, A., Wardhana, N.M.: Topologic, https://topologic.app, Last accessed: April 03, 2020, (2019).
20. Jabi, W., Soe, S., Theobald, P., Aish, R., Lannon, S.: Enhancing parametric design through non-manifold topology. Des. Stud. 52 96–114 (2017).
21. Laugwitz, B., Held, T., Schrepp, M.: Construction and evaluation of a user experience questionnaire. In: Holzinger, A. (ed.) Proceedings of the 4th Symposium of the Workgroup Human-Computer Interaction and Usability Engineering of the Austrian Computer Society on HCI and Usability for Education and Work. pp. 63–76. Springer, Berlin, Heidelberg (2008).
22. Lee, S.-U., Roh, M.-I., Cha, J.-H., Lee, K.-Y.: Ship compartment modeling based on a non-manifold polyhedron modeling kernel. Adv. Eng. Softw. 40 (5), 378–388 (2009).
23. Love, P.E.D., Edwards, D.J., Han, S., Goh, Y.M.: Design error reduction: toward the effective utilization of building information modeling. Res. Eng. Des. 22 (3), 173–187 (2011).
24. Masuda, H., Shimada, K., Numao, K., Kawabe, S.: A mathematical theory and applications of non-manifold geometric modeling. In: Krause, F.-L. and Jansen, H. (eds.) International Symposium on Advanced Geometric Modelling for Engineering Applications. North-Holland, Berlin (1989).
25. McCullough, M.: Abstracting craft: the practiced digital hand. MIT Press, United States (1997).
26. Open CASCADE: Open CASCADE, http://www.opencascade.com, Last accessed: May 15, 2018, (2018).
27. Qiuchen Lu, V., Parlikad, A.K., Woodall, P., Ranasinghe, G.D., Heaton, J.: Developing a dynamic digital twin at a building level: using Cambridge campus as case study. In: DeJong, M., Schooling, J., and Viggiani, G. (eds.) International Conference on Smart Infrastructure and Construction 2019 (ICSIC). pp. 67–75. ICE Publishing (2019).
28. Roberts, A., Marsh, A.: ECOTECT : environmental prediction in architectural education. In: Architectural Information Management: 19th eCAADe Conference. pp. 342–347. , Helsinki, Finland (2001).
29. Sabry, H., Sherif, A., Rakha, T., Fekry, A.: Integration of daylighting simulation software in architectural education. In: Tizani, W. (ed.) The 13th International Conference on Computing in Civil and Building Engineering. Nottingham University Press (2010).
30. Schön, D.A.: The reflective practitioner: how professionals think in action. Basic Books (1984).
31. Spatial Corporation: 3D ACIS modeler, https://www.spatial.com/products/3d-acis-modeling, Last accessed: March 03, 2020, (1989).
32. Voloshin, V.I.: Introduction to graph and hypergraph theory. Nova Kroshka Books (2013).
33. Weiler, K.J.: Topological structures for geometric modeling. Rensselaer Polytechnic Institute (1986).
34. Woodbury, R., Gün, O.Y., Peters, B., Sheikholeslami, M. (Roham): Elements of parametric design. Taylor & Francis, Inc. (2010).