A Web Serverless Architecture for Buildings Modeling

Enrico Marino*, Danilo Salvati†, Federico Spini*, Christian Vadalà†
*Department of Engineering, Roma Tre University, Rome, Italy
Email:{marino,spini}@ing.uniroma3.it
†Department of Mathematics and Physics, Roma Tre University, Rome, Italy
Email:{salvati,vadala}@ing.uniroma3.it

Abstract—**The motivations of relentless migration of software products toward services accessible via Web must be sought in the undeniable benefits in terms of accessibility, usability, maintainability and spreadability granted by the Web medium itself. It is the case of office suites, beforehand thought as resilient desktop applications, nowadays made available as Web applications, often equipped with real-time collaboration features and with no need for the user to explicitly install or upgrade them anymore. Although it could not be easy to envisage a Web-based graphic application due to its inherent complexity, after the recent and significant enrichment of the HTML5 APIs, a few first attempts appeared online in the form of vectorial drawing collaborative editors or VR oriented interior design environments. This paper introduces an effective Web architecture for buildings modeling that leverages the serverless pattern to dominate the developing complexity. The resulting front-end application, powered by Web Components and based on unidirectional data flow pattern, is extremely customizable and extendible by means the definition of plugins to augment the UI or the application functionalities. As regards the modeling approach, it offers (a) to model the building drawing the 2D plans and to navigate the building in a 3D first person point of view; (b) to collaborate in real-time, allowing to work simultaneously on different layers of the project; (c) to define and use new building elements, that are furnitures or architectural components (such as stairs, roofs, etc.), augmenting a ready to use catalog. This work suggests a path for the next-coming BIM online services, matching the professionals collaboration requirements typical of the BIM approach with the platform which supports them the most: the Web.**

1. Introduction

Nowadays we are seeing a relentless migration of software products toward services accessible via the Web medium. This is mainly due to the undeniable benefits in terms of accessibility, usability, maintainability and spreadability granted by the Web medium itself. Nevertheless these benefits don't come without a cost: performance and development complexity become major concerns in the Web environment.

In particular, due to the several introduced abstraction layers it is not always feasible to "port" a desktop application into the Web realm, an aspect to be taken into account even for the relevant hardware differences among all the devices equipped with a Web Browser. It can be even more arduous to tackle the inherent distributed software architecture (a client/server one at least) induced by the Web platform.

Nevertheless increasingly rich and complex Web applications began to appear, supported by the enriched HTML5 APIs, which thanks to the WebGL [1] (which enables direct access to GPU), Canvas [2] (2D raster APIs) and SVG [3] (vectorial drawing APIs), has paved the way for the entrance of Web Graphic Applications.

In this work we report about our endeavor toward the definition of a Web based buildings modeling tool which overcomes the aforementioned performance and development difficulties relying on a unidirectional data flow design pattern and on a serverless architecture [4], respectively.

A serverless architecture, on the contrary of what the name may suggest, actually employs many different specific servers, whose operation and maintenance don't burden on the project developer(s). These several servers can be seen as third party services (typically cloud-based) or functions executed into ephemeral containers (may only last for one invocation) to manage the internal state and server-side logic. Realtime interaction among users jointly working on the same modeling project, is for example achieved via a third party APIs for remote users collaboration.

The tool user interface, entirely based on **web components pattern**, has been kept as simple as possible: the user is required to interact mainly with two-dimensional symbolic placeholders representing parts of the building, thus avoiding complex 3D interactions. The modeling complexity is thus moved from the modeler to the developer which fills out an extendible **catalog** of customizable *building elements*. The modeler has only to select the required element, place and parametrize it according to the requirements. It is obvious that a large number of building elements has to be provided to ensure the fulfillment of the most modeling requirements.

The remainder of this document is organized as follows. Section 2 provides an overview of related work. Section **??** presents adopted architectural solutions. Section 5 contains results, some conclusive remarks and figures future developments.

2. Related work

In this section we highlight some remarkable experiences aligned with the aim of our project. There are plenty of Desktop applications worth to be mentioned and analyzed, but in the following we deliberately focus on Web based works.

Shapespark[1] offers a web viewer of remarkable quality that allow the user to move inside a synthetic 3D indoor environment. Modeling phase is served in the form of plugins for different Desktop proprietary solutions.

Playcanvas[2] is a complete and powerful web based game creation platform which offers an integrated physical engine and a whole set of functionalities to support modeling. Although powerful and relatively simple to use, it doesn't focus on buildings modeling.

Floorplan[3] has been developed by Autodesk specifically for the architectural field, and for indoor renewal projects in particular. It is a 2D modeling tool which offer also a 3D walk-through mode.

Spini et al. [5] introduced a Web modeling and baking service for indoor environments. The modeling tools exposes a 3D interaction the user may not be accustomed to, an hitch we tried to outflank by avoiding 3D modeling interaction and let the user only face a "metaphoric" 2D interface.

3. Application Experience

blabla

3.1. User Interface

blabla

3.2. Building Elements Catalog

blabla

4. Serverless Architecture

blabla

4.1. Centralized Application State

blabla

4.2. Component Based UI

blabla

1. https://www.shapespark.com/
2. https://playcanvas.com/
3. http://www.homestyler.com/floorplan/

5. Conclusions

Now we can see the results obtained from the adoption of the techniques we have studied in the previous sections. The uniflow pattern allowed us to identify a consistent and global state which describes the state of the project and the business logic. Moreover, in conjunction with the adoption of a serverless architecture, it led to a slim architecture for the collaboration. Using the uniflow pattern with the Virtual DOM, permitted us to automatically render only changes to the state without complex optimization algorithms, like the one based on differences and patches used for the 3D rendering of the building model. In addition the system is extensible with the addition of geometric elements, in fact the user can register external services to add new ones. In conclusion, use of the Web Components allow us to expose our software in a modular way, improving maintainability.

5.1. Future work

The system could be expanded thanks to the modularity given by the serverless paradigm. As we already have a good user experience, to help the adoption of this application in other modeling contexts, we should improve the catalog adding new building elements

Acknowledgments

The authors would like to thank Stefano..... and Prof....

References

[1] D. Jackson, "WebGL Specification," Khronos, Khronos Recommendation, Oct. 2014, https://www.khronos.org/registry/webgl/specs/1.0.3/.

[2] J. Munro, J. Mann, I. Hickson, T. Wiltzius, and R. Cabanier, "HTML Canvas 2D Context," W3C, W3C Recommendation, Nov. 2015, http://www.w3.org/TR/2015/REC-2dcontext-20151119/.

[3] D. Jackson, E. Dahlström, J. Ferraiolo, A. Grasso, C. McCormack, P. Dengler, J. Fujisawa, D. Schepers, C. Lilley, and J. Watt, "Scalable Vector Graphics (SVG) 1.1 (Second Edition)," W3C, W3C Recommendation, Aug. 2011, http://www.w3.org/TR/2011/REC-SVG11-20110816/.

[4] M. Roberts, "Serverless Architectures." [Online]. Available: http://martinfowler.com/articles/serverless.html

[5] F. Spini, E. Marino, M. D'Antimi, E. Carra, and A. Paoluzzi, "Web 3D Indoor Authoring and VR Exploration via Texture Baking Service," in *Proceedings of the 21st International Conference on Web3D Technology*, ser. Web3D '16. New York, NY, USA: ACM, 2016, pp. 151–154. [Online]. Available: http://doi.acm.org/10.1145/2945292.2945309

[6] C. A. Ellis and S. J. Gibbs, "Concurrency Control in Groupware Systems," *SIGMOD Rec.*, vol. 18, no. 2, pp. 399–407, Jun. 1989. [Online]. Available: http://doi.acm.org/10.1145/66926.66963

[7] J. Hopcroft and R. Tarjan, "Algorithm 447: Efficient Algorithms for Graph Manipulation," *Commun. ACM*, vol. 16, no. 6, pp. 372–378, Jun. 1973. [Online]. Available: http://doi.acm.org/10.1145/362248.362272

[8] J. Rotolo, "The Virtual DOM vs The DOM." [Online]. Available: http://revelry.co/the-virtual-dom

[9] P. Bryan and M. Nottingham, "JavaScript Object Notation (JSON) Patch," Internet Requests for Comments, RFC Editor, Tech. Rep. 6902, April 2013.