Loosely Coupled Approach for Web-Based Collaborative 3D Design

Caroline Desprat, Benoît Caudesaygues, Hervé Luga, Jean-Pierre Jessel

Doctoral Symposium: Loosely Coupled Approach for Web-Based Collaborative 3D Design

Caroline Desprat*
University of Toulouse
118 route de Narbonne
Toulouse, France
desprat@irit.fr

Benoît Caudesaygues
benoit.caudesaygues@gmail.com

Hervé Luga
University of Toulouse
118 route de Narbonne
Toulouse, France
luga@irit.fr

Jean-Pierre Jessel
University of Toulouse
118 route de Narbonne
Toulouse, France
jessel@irit.fr

ABSTRACT

Recent advances in Web 3D technology have opened a wide area for Collaborative Virtual Environments (CVE). While CVE are often viewed in a concurrency context, they need to provide a satisfying experience in terms of consistency, latency and recovery. Because (i) Event-Driven architectures (EDA) are well-suited for distributed application and (ii) traditional communication architecture (client-server) can be limited in such situations, this paper presents a loosely-coupled approach combining event sourcing with a hybrid communication architecture. This model aims to ensure a strong versioning system and resource availability for collaborative 3D object manipulation in a web browser. To evaluate acceptance of our system, we conducted a user study on groups of users working simultaneously on 3D cooperative assembly tasks. The results detail the users' involvement evolution, qualitative appreciations of the system's usability and the collaborative features.

CCS CONCEPTS

•Human-centered computing →Collaborative content creation; •Software and its engineering →Peer-to-peer architectures; •Networks → *World Wide Web (network structure);*

KEYWORDS

Event-Driven Architecture; Event sourcing; Web 3D; WebRTC; P2P

DOI: http://dx.doi.org/10.1145/3093742.3093905

*Caroline Desprat is a PhD student at University of Toulouse

1 INTRODUCTION

Web-based collaboration represents the *status quo* for many different systems including CVE [7]. There are still open challenges, especially concerning 3D design, such as designing a distributed collaborative architecture for development, analysis and evaluation of collaborative sessions. In the area of distributed application architectures there is an architectural style called event sourcing (ES) that shares the same ideas as 3D collaborative design in industrial context: preserving metadata (non-graphical attributes), design intent data (history tree) and application data over product lifecycle [12]. Recently, ES has been gaining attention within distributed and web-based systems because its style, as opposed to state-based systems, does not modify state by altering values. Instead, it records the sequence of events which trigger state transitions [4].

Following the works of Desprat et al. [3] which presents a light event-based solution for 3D CVE using an hybrid architecture strongly relying on the server, we propose in this study an upgrade on their communication architecture by making peers totally autonomous to allow higher network availability and scalability. Our user study includes cooperative tasks to fulfill two purposes: it supports our beliefs that event-based architecture for collaboration is useful for tracking evolution on 3D scenes and it produces a set of the traces of users thanks to event recording, allowing us to analyse the collaborative sessions. The event-driven architecture, totally deported on peers, is the core of the model evaluated here, showing compatibility of such architecture with 3D collaborative design.

This paper is structured as follows. First, an overview of related work introduces 3D web-based collaboration and related concepts for event-driven architectures. Then, Section 3 describes our model including the client event-based model for 3D design and our communication architecture. Section 4 details the experiment to validate the usability, acceptance and reliability of our system. Finally, Section 5 presents the results and we conclude in Section 6.

2 RELATED WORK

The mechanics of collaborations apply to 3D modeling through the small-scale actions and interactions that group members must carry out in order to collaborate within a shared workspace [10]. 3D web-based collaboration relies on three factors: (i) communication; (ii)

the rapid and fine-grained updates; and finally, (iii) the embodiment of the user so other can see what he/she is working on.

In one hand, recent developments in cloud technologies raise the need for interoperability in terms of devices and communication architectures. WebRTC [6] is a W3C standard published as a working draft that allows peers to connect and exchange audio, video and data with each other. Although WebRTC enables P2P connections, it requires a server (usually a WebSocket server) for both signaling peers and coping with NAT/firewalls. The signaling mechanism coordinates the network metadata exchange to bootstrap a peer connection. Hybrid architectures (client-server and P2P via WebRTC) have appeared in various collaborative domains like 3D assets delivery [8] and 3D design [1, 3] encouraging the reduction of the cost of collaboration, communication and processing in heterogeneous environments.

In the other hand, the benefits of EDA have been proven in distributed architectures with loose-coupled and non-monolithic design, openness and scalability properties as illustrated by Hohpe [5]. The groupware system presented in [11] focuses on distributed event-based awareness in P2P integrating requirements among which are: (i) generic representation for events, (ii) distributed data, (iii) mechanisms for reducing the overhead caused by event processing, and (iv) handling of peers' dynamics (join and leave) while taking care of data synchronization and consistency on each peer. Using structured P2P networks in EDA (transmitting an event only to those neighbours situated on the path towards its subscribers) can greatly reduce traffic when it comes to small teams because events need to be transmitted to few peers [2]. Considering data management, event sourcing (ES) [4] consists of ensuring every state change of an application is captured in an event object. It uses an append-only store to record the sequence of state-modifying events (stream). The consistency of transactional data issued from ES provides data traceability. In distributed EDA, ES is often used with the Command Query Responsibility Segregation (CQRS) pattern to manage concurrency conflicts and to capture user intention [13]. CQRS intends to limit access to domain objects by exposing two models: the command model handles operations that update entities' state while the query model provides read-only access to object states.

3 MODEL

In our collaborative system, users concurrently assemble parts of a 3D model on their web browser client. Collaboration is established by synchronising users' edits in near-real time from an authoritative instance. Our prototype supports high level manipulating of 3D triangle meshes. Our model relies on three aggregates: Scene, Mesh, Geometry (this paper focuses on the former). An aggregate is a cluster of domain objects that can be treated as a single item with its own version numbering. In a Scene, the meshes composing it are separate entities, but it is useful to treat the Scene with its consubstantial meshes as a single aggregate.

3.1 Event-based model for 3D design

Each 3DEvent instance includes CQRS and ES patterns. Usually implemented in a client-server architecture, we adapted CQRS and ES to a P2P architecture by totally deporting it on a peer node (3DEvent instance). We distinguished two types of 3DEvent instance: Web instances and Server instances. A Web Instance is dedicated to a user for creating, manipulating 3D objects via the user interface integrated in a web browser. In the model, it is the only instance that produces events, consume and transmit events. A Server instance, oppositely to the Web instance, it does not produce any event but consumes and transmits events. It is the interface between Web instances and the database. The Server instance is fully integrated in the P2P network.

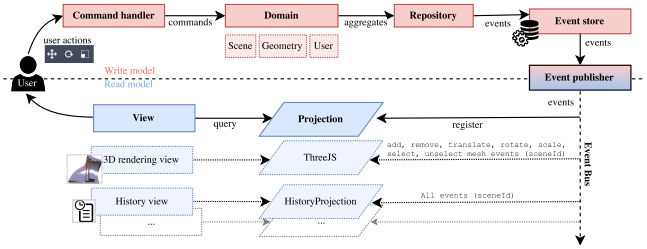


Figure 1: Web instance: CQRS inside the web browser

Figure 1 shows the workflow on a 3DEvent Web instance: from a user's action to its visualisation. On the write model (upper side), when the user triggers a command from the UI, it is handled by the domain in order to be validated (based on predefined rules). If validated, the domain produces/modifies the aggregate concerned. These modifications are wrapped into events and sent to the Event Store to be processed and stored. Then, the Event publisher passes them to the Event Bus on which projections are hooked. Finally, views (UI components) are updated by querying their projection.

The Event Store is composed of the event stream manager (ESM) and the network bridges (NB). The ESM stores locally data as event streams (array of indexed events). When the event store receives new events from the current instance, the consistency of version is checked by comparing the expected version (exposed in the metadata of the event) and the actual version (last index of the aggregate's event stream). If the two versions are equal, the event is pushed into the event stream, otherwise a concurrency exception is raised and handled according to business rules. When stored in the ESM, the event is published. A NB is the component that carries out the WebRTC connection with other peers. The NB is the interface between the ESM and an external event stores. The progressive rendering of the scene is assured by the 3D rendering projection capable of reordering events. Using this method, visual feedback can be displayed to replace missing geometries (e.g. bounding boxes) while a scene is loading. The UI is a 3D environment with standard high level object manipulation (translate, rotate, scale). The selection/deselection visual feedback shows the object a user is working on. The user selects and works on a ghost clone of the object until he/she is satisfied of the modification and then validates the modification. The validation triggers the command that goes through the CQRS workflow and the ghost is deleted. This solution allows the user to see the previous position of the original object and also warns for concurrent selection. During collaboration, the selection does not relies on the lock mechanism because it can lead to starvation issues. Instead, we preferred a non blocking solution

where the last arrived event is applied. When an object is selected by multiple users at the same time, no concurrency issue is raised.

3.2 Communication architecture

Our model uses an hybrid communication architecture bringing advantages of both client-server and P2P in a web-based CVE (Figure 2). This model answers both the expectation of rapid data dissemination to the collaborators, and the need for long-term persistence of collaborative session and workspace evolution. The system has the inherent ability to scale-up by just requiring presence of collaborators (their Web instances) and the presence of any number of Server instances (assuring the resource availability using total replication). The architecture is composed of 3DEvent instances (relying on WebRTC connection for synchronisation), the Instance Manager (a WebSocket server) and the DB (the long-term persistence).

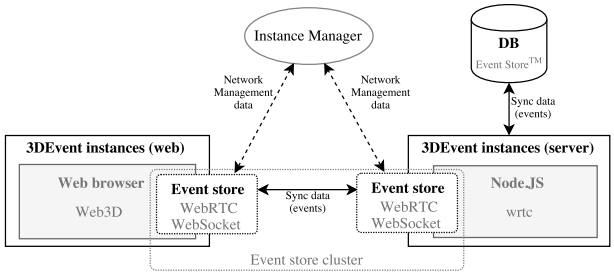


Figure 2: Communication architecture

3.2.1 Synchronisation mechanism. The synchronisation mechanism consists, for a peer, of asking all connected peers for their updates. The exchange is done through the WebRTC connection hosted by NB of each Event Store. The NB receives a metadata message describing the resources of the remote ESM. The ESM retrieves the differences (missing and incomplete event streams) between its event streams and the remote event streams. From this diffing, the NB send messages over the WebRTC connection containing events according to the diffing produced. This mechanism is done until reaching the same version number for each event stream. It is repeated as many time as needed (new peer connection, recovery).

For instance, a 3DEvent instance (*idA*) joins the network when other 3DEvent instances are already present in the P2P network. The join action is executed when a user sends its connection settings at start. This action adds the peer to the list and returns the list of peers to connect with *ids*. For each peer *idB* of *ids*, *idA* uses the signaling mechanism (offer/answer) triggered by the instantiation of a NB in the Event Store of *idA* then the *idB*'s one. To be synchronous after these (asynchronous) exchanges, *idA* and *idB* exchange metadata concerning their respective ESM situation and synchronise. To get a resource, *idA* can ask different instances. Once every connection is well established, the real-time collaboration is ready.

3.2.2 Long-term persistence. The long-term persistence stores the immutable events. The Server instances use the long-term persistence for synchronising their ESM at connection time. The persistence is centralised and considered as the authoritative source of data for synchronisation. Inherent functional programming benefits coming from ES reduce the risk of errors.

3.3 Implementation

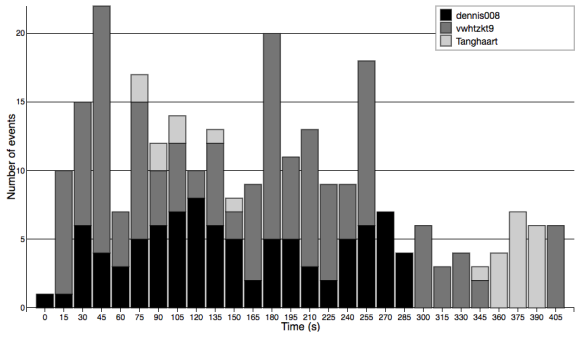
The application presented in this paper is fully implemented in TypeScript. On Web instances, the application is provided as a Single Page App to avoid page reloading at each action. The 3D visualisation is done using the Three.JS library. The data are transmitted using RTCDatachannel API of WebRTC. The Server instances are based on Node.JS and provide HTML/CSS contents. The WebRTC API on Node.JS is enabled through the node-webrtc npm module. The long term persistence relies on the EventStore® library.

4 USER STUDY

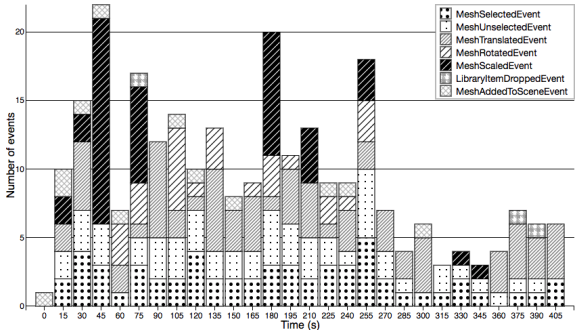
This experiment tried to replicated a realistic collaboration between participants working remotely in the same city. This user study intends to highlight the reliability of the 3DEvent application. We wanted to observe participants' behaviours in quantitative (monitoring) and qualitative (questionnaire) terms executing collaborative tasks in our system. Participants had to assemble in a web-browser the different parts of a model using 3DEvent application and features to match a final given assembly. The experiment was conducted on 6 participants. The experiment protocol allocates them in different groups. They operated on distinct networks with good internet connection (at least 20Mb/s). The participants were master/PhD students familiar with computer science (not 3D modeling necessarily). They were allowed to communicate between each-other by chat. The experiment contains three phases: (1) Trial phase, (2) Solo phase, (3) Collaboration phase. (1) Participant is training for 5-10 minutes on the application to become acquainted with the interface and features. (2) Participant does an assembly of 10 parts model (65k triangles, 5MB). (3) A group of participants realises two assemblies: (i) a 10 parts model and (ii) a 16 parts model. The collaboration phase was conducted 6 times: 3 with groups of two participants and 3 with groups of three participants. To avoid learning-related biases, different models with similar characteristics (number of parts and triangles) has been presented. For each phase, the application first loads the model parts in the object library of each participant. For each trial and each participant, we recorded the completion time, the number of events and the timestamp for each event to observe if the number of collaborators eases the task completion (efficiency and speed). Time recording started at the first click in the scene and stops when the group says the task is complete (last click timestamp). Apart from the collected data, participants were given a qualitative questionnaire to evaluate the usability of the system and their collaborative involvement [9].

5 RESULTS AND DISCUSSION

Over the different collaborative sessions, the applications produced hundreds of events (approx. 300 per session). We went through many network instabilities that were due to the youth of the WebRTC standard. During the user study, all participants had a very good internet connection (>20Mb/s), one participated on 4G. Thanks to the user traces, we are able to retrieve which (Figure 3a) event has be done by whom (Figure 3b). To have an overview of the possibilities, we analysed a collaborative session between three users. Figure 3 shows the different aspects of the recorded session on the *living room* model (16 parts, 200k triangles, 9MB). At the beginning, we observe a lot of mesh additions. Only one user added



(a) By users



(b) By event types

Figure 3: Summary of a collaborative session over time

a mesh through the dropping feature. The amount of selection and deselection is quite similar. Sometimes there is no deselection triggered (selection changed without deselection in between). When starting the session, the users interacted heavily (until 20 events in 15s). Then, we observe that the three users interacted together during few minutes before one of them leaves and comes back (info gathered from the user aggregates not showed here). At the end, we observe a decreasing number of events, meaning that the users are finishing the task. We also see the users' involvement through the frequency of their contribution and the type of each generated event. In all experiments, the goal was reached in about the same time (10-15 minutes). The purpose of the application fits the 3D cooperative assembly context well because participants succeeded rapidly. From non interactive to real-time, participants qualified the application as near-real-time. The general satisfaction of the experience and the collaboration satisfaction shows that participants enjoyed the user experience of 3D collaborative modeling in a web browser. When we asked if the the number of users improve both efficiency and speed to complete the tasks, the participants generally agreed. During one of the experiments, we had a latency of 10 seconds, but despite it, participants said that the latency did not affect the collaboration. Few conflicts were raised between users (concurrent selection) but rapidly solved by the system.

In the questionnaire, participants globally expressed the fact that they executed collaborative tasks more quickly and more efficiently than solo tasks. The ease of use and the simplicity of the interface were pointed out by several users. Network stability sometimes led to frustration for some users. However, they found the collaborative consistency and recovery acceptable. The data distribution worked well, allowing them to cooperate efficiently.

6 CONCLUSION

This paper presents an original event-driven architecture for 3D web-based modeling in collaboration. The event-driven approach exposes several benefits in this context: the history of modifications is very detailed with small memory footprint and exposes user's intent in action under business constraints. We propose an implementation for a distributed and asynchronous cluster of Event Stores over the P2P network. We fully integrated servers in the P2P network and positioned users as data distribution contributors. The user study shows encouraging results in terms of responsiveness and usability of the system. The application responded well concerning rendering and distribution performances for a web-based 3D CVE. In the future, we will aim to stabilise the network by improving compatibility with non WebRTC clients. The robustness and the (global) synchronization mechanisms reliability used in this model are not established. Improving these two parameters would ensure better consistency in long-term use of such model. Finally, granularity refinement would be interesting to embrace to get finer details of 3D collaborative modeling such as collaboration pattern detection through complex event processing.

REFERENCES

[1] Haroula Andrioti, Andreas Stamoulias, Kostas Kapetanakis, Spyros Panagiotakis, and Athanasios G Malamos. 2015. Integrating WebRTC and X3DOM : Bridging the Gap between Communications and Graphics. In *Web3D '15: Proceedings of the 20th International Conference on 3D Web Technology*. 9–15.

[2] Valentin Cristea, Florin Pop, Ciprian Dobre, and Alexandru Costan. 2011. Distributed architectures for event-based systems. *Studies in Computational Intelligence* 347 (2011), 11–45. DOI: http://dx.doi.org/10.1007/978-3-642-19724-6{_}2

[3] Caroline Desprat, Jean-Pierre Jessel, and Hervé Luga. 2016. 3DEvent: A Framework Using Event-Sourcing Approach For 3DWeb-Based Collaborative Design in P2P. In *Proceedings of the 21st International Conference on Web3D Technology - Web3D '16*. 73–76. DOI: http://dx.doi.org/10.1145/2945292.2945310

[4] Martin Fowler. 2003. *Patterns of Enterprise Application Architecture*. Vol. 23. Addison-Wesley Longman Publishing Co., Inc. 415 pages.

[5] Gregor Hohpe. 2006. Programming Without a Call Stack – Event-driven Architectures. *Enterprise Integration Patterns* (2006).

[6] Cullen Jennings, Ted Hardie, and Magnus Westerlund. 2013. Real-time communications for the web. *IEEE Communications Magazine* 51, 4 (2013), 20–26. DOI: http://dx.doi.org/10.1109/MCOM.2013.6495756

[7] Chris Joslin, Thomas Di Giacomo, and Nadia Magnenat-Thalmann. 2004. Collaborative virtual environments: From birth to standardization. *IEEE Communications Magazine* 42, 4 (2004), 28–33. DOI: http://dx.doi.org/10.1109/MCOM.2004.1284925

[8] Timo Koskela, Arto Heikkinen, Erkki Harjula, Mikko Levanto, and Mika Yliant-tila. 2015. RADE : Resource-aware Distributed Browser-to- browser 3D Graphics Delivery in the Web. *IEEE Wireless and mobile* (2015), 500–508.

[9] J.R. Lewis. 1995. IBM Computer Usability Satisfaction Questionnaires: Psychometric Evaluation and Instructions for Use. *International Journal of Human-Computer Interaction* 7, 1 (1995), 57–78. DOI: http://dx.doi.org/10.1037/t32698-000

[10] David Pinelle, Carl Gutwin, and Saul Greenberg. 2003. Task Analysis for Groupware Usability Evaluation: Modeling Shared-Workspace Tasks With the Mechanics of Collaboration. *ACM Transactions on Computer-Human Interaction (TOCHI'03)* 10, 4 (2003), 281–311. DOI: http://dx.doi.org/10.1145/966930.966932

[11] Fatos Xhafa and Alex Poulovassilis. 2010. Requirements for distributed event-based awareness in P2P groupware systems. *24th IEEE International Conference on Advanced Information Networking and Applications Workshops, WAINA 2010* October (2010), 220–225. DOI: http://dx.doi.org/10.1109/WAINA.2010.112

[12] Xun Xu. 2011. *Integrating advanced computer-aided design, manufacturing, and numerical control: principles and implementations, by X. Xu.* Vol. 49. 3425–3426 pages. DOI: http://dx.doi.org/10.1080/00207543.2010.501547

[13] Greg Young. 2009. Code Better. (2009). http://codebetter.com/gregyoung/