# Transformation from the Heavy Desktop Client to the Lightweight Web Application

Richard BELAN*

*Slovak University of Technology in Bratislava*
*Faculty of Informatics and Information Technologies*
*Ilkovičova 2, 842 16 Bratislava, Slovakia*
xbelanr1@fiit.stuba.sk

**Abstract.** Recently the trend is to create applications which are always available online. This leads to expansion of Web technologies and creation of applications directly for Web. With HTML5 and CSS3 there are numerous possibilities to create complex graphical applications on the web. In this paper we propose a way how to migrate heavy graphical applications from desktop to the Internet. This paper also describes the way we took to migrate our graphical desktop prototype and compares some of the current technologies and options available for migration of desktop applications.

## 1 Introduction

Modelling and designing large, corporate software with complex requirements needs improvements in graphic visualization for easier creation of models, better understanding of diagrams and better collaboration between designers and designer teams through various countries, time zones, departments and divisions in cooperative creation of applications their models and architecture together.

Unified Modelling Language (UML) is a standardized language used in software modelling for describing architecture and functionality of software systems. There are a number of available tools, programs and large applications for creation of UML diagrams with 2D space support. Three-dimensional space brings many new important features for UML diagram creation. Minimization of intersections, reduction of diagram complexity and allowing visualization of complex diagrams in advanced up-to-date graphics for achieving easier to read schemas of large models with decomposing the structure into specific modules, components, objects, types, patterns, authors and time are all of the best benefits of adding another spatial dimension into software modelling and moving UML diagramming from two-dimensional space to three-dimensional space.

With rapid progress in technologies in past years and with everything being on Web, we had to adjust our solution to this growth and migrate our solution from desktop client to a Web Application. This paper brings the journey needed for migrating a graphic application from desktop to web and all the trouble encountered.

---

* Master degree study programme in field: Software Engineering
Supervisor: Dr. Ivan Polášek, Institute of Informatics and Software Engineering, Faculty of Informatics and Information Technologies STU in Bratislava

## 2 Related work

Although there is no commercial three-dimensional UML modelling software, there are some alternatives available. These existing prototypes try to use the third dimension in numerous ways. The three main methods are dividing models and diagrams into layers, using advanced three-dimensional animations [1] and having three-dimensional object representation with possible alternative shapes and forms [3]. H. Koike proposed the first three-dimensional software visualization prototype named VOGUE [1,2], which used third dimension to visualize message sending between processes. Already with adopting UML as a standard for two-dimensional modelling in 1997 many three-dimensional alternatives were proposed. One of the first were J. Gil and S. Kent who first proposed the idea of layers and division of information [4] and they created a three-dimensional notation for UML class diagram and a nested-box diagram [4].

The most complex solution [5] was made by Paul McIntosh and it was his three-dimensional UML state machine diagram. P. McIntosh extensively studied the benefits of third-dimension in software modelling and visualization in comparison with traditional two-dimensional UML approaches [5, 6]. His solution was compatible both with ISO based X3D standards and also with UML standards therefore he named his solution X3D-UML [6]. His approach is best visible on hierarchical state diagrams as his three-dimensional state diagrams uses movable hierarchical layers with ability to apply filtering. Numerous information are not visible on traditional 2D diagrams therefore the person must think outside the diagram. With using third dimension there are new possibilities to show more information on screen and reduce the need of thinking out of diagram [6]. In addition to this research he also studied methodology sequential evaluation used in 3D user interfaces.

Another approach which extensively studied the benefits of third dimension and created a prototype for three-dimensional visualization and software modelling was GEF3D [7]. Their first approach is to visualize connections between two-dimensional diagrams in three-dimensional space [6]. This approach uses the idea of layering information and can be used for model driven development where the models are chained. The second approach is to project diagrams into sides of a cube and arrange connections between diagrams through the cube. GEF3D is so far still in the Eclipse incubator and not released officially.

American vision scientist I. Biederman known for his RBC theory (Recognition-by-components theory) [8] in which he created a set of three-dimensional objects of primitive shape called Geons. He claimed that these objects and composed objects made solely of these objects are easier for human recognition and remembering than any other type of objects. I. Pourang proposed a way of creating diagrams only with Geon shaped objects and also made a study [3] where students remembered better and also could recognize easier the Geon shaped diagrams instead of the standard UML diagrams. Pourang claimed that these diagrams are closer to human perception and by using these shapes the person can recreate the mental map much easier. Although using this method for large, complex diagrams was never proved and tested.

## 3 Our approach

To stay “up-to-date” with current technological trends and application needs to be always upgraded. There are situations when an additional upgrade is impossible or near impossible and migrating the whole project from one technology to another is a better choice. Now when everything is online and easily accessible there is a need for migrating large desktop client applications to Web. Web offers many advantages opposite to desktop applications. The main advantages are:

- Easier accessibility – Sending access link instead of complete software with installation manuals.
- Smaller client-side client – Compute most of the application logic on backend Server

Considering these advantages and our opportunities in upgrading our existing desktop client we chose to wrap everything usable from our application, reuse it and rewrite it for Web.

## 3.1 3D-UML desktop client

Our desktop client is a robust graphic application written in C++ and using OGRE3D graphic engine. It is a tool for creating 3D UML diagrams. Currently it supports Sequence, Class and Activity diagrams. Each of the diagrams has their own UML Metamodel created for 3D space like the Metamodel for Class diagram [11] on Figure 1. Metamodels for supported diagrams are compatible with UML standards as they were created from OMG UML Superstructure and Infrastructure [9]. Class diagram supports Force-Directed algorithms [12] like weighted Fruchterman-Rheingold and Sequence diagram is using modern combined fragments for alternative and parallel scenarios [14].

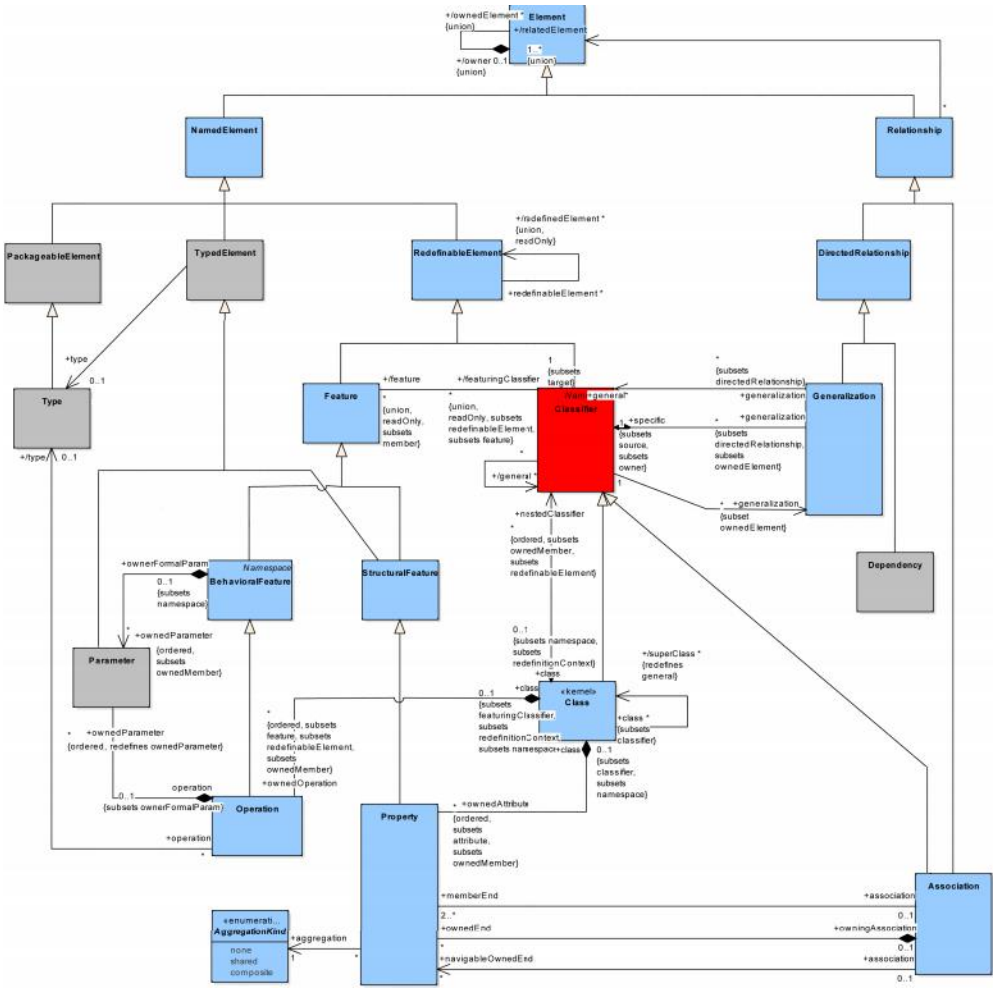


*Figure 1. Class diagram Metamodel.*

## 3.2 MMVCC architectural structure

Change of Architectural structure was the first big upgrade to our prototype. The former MVC architectural style was not enough due to our extensive use of 3D elements and backward

compatibility with CASE system by saving the elements as 2D too. This way we ended up having two models, one for the 2D elements and another for 3D elements. This was the first part of our transformation from MVC to MMVCC. The second part consisted of dividing the controller. We divided the universal controller for both 2D and 3D elements into two separate controllers. That led to finishing our new architectural style. MMVCC stands for Model-Model-View-Controller-Controller.

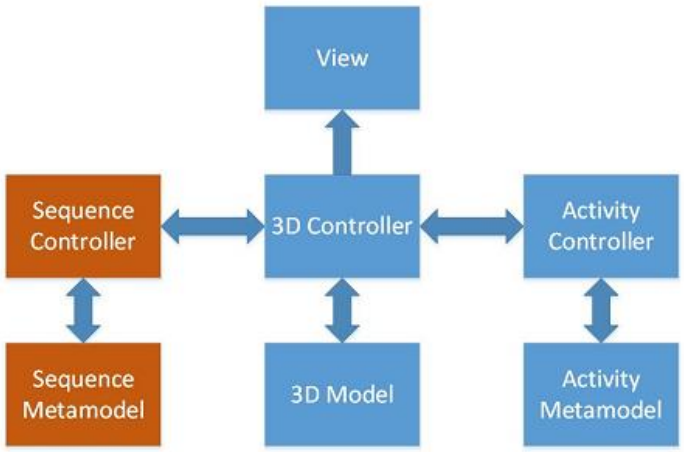


*Figure 2. MMVCC Architecture.*

## 3.3 Transformation of large desktop client to Web

When transforming a desktop application, it is important to determine the needs at first because there is a difference between transforming a simple form application and an advanced application using a lot of enhanced graphics. The backend is not discussed in this paper because there are many Server-Side JavaScript based languages (e.g. NodeJS), which perfectly can do everything that all main desktop languages like C++, C# or Java are capable of.

### 3.3.1 Transforming form applications

Transforming form applications is the easier of the two. There are mostly no problems encountered during the process as the current technologies for form-like applications on the Web are well developed and supported [13]. With HTML5 and CSS3 it is no more a problem. Using user interface Frameworks like Bootstrap the output display of the Web applications might be even better than the desktop client due to features like responsive design, high flexibility, ease of use and many reusable templates. Desktop form applications are using mainly 2D technologies, and 2D technologies are well developed and standardized for web therefore migrating 2D form applications from desktop to web is no longer a problem.

### 3.3.2 Transforming graphic applications

Graphic applications are much harder to transform and migrate than form applications. There is not yet a standardized way for migrating graphical desktop applications to web. In many cases transforming them to pure HTML and CSS is almost impossible. Although there are rare experiments, when even games were developed using advanced HTML5 and CSS3 hacks. These solutions tend to be ineffective and lagging [11]. This is due to current situation that 3D is not yet standardized for web and there are three main streams in developing such applications, these are HTML canvas, WebGL and CSS 3D transformations.

HTML5 canvas uses only 2D transformations and the 3D effects have to be self-done which leads to missing hardware optimized graphic rendering. WebGL is a JavaScript API for rendering interactive 2D/3D computer graphics. It is using HTML5 canvas and OpenGL ES as its base. Frameworks based on WebGL and WebGL itself is capable of creating advanced graphic with complex elements using hardware accelerated graphic rendering, but with using WebGL we lose the features of HTML features as DOM manipulation, input tags so it is important to choose WebGL only when needed. CSS 3D transformations are getting more popular and their functionality is being further expanded and in future it might be the universal way for nay graphic application. Now it has limitations on advanced graphic effects and complex geometrically shaped objects.

Difference between the WebGL approach and the pure CSS 3D transformations approach can be seen on Figure 3. There we see how the WebGL can be displayed together with HTML/CSS elements, but it itself does not contain these elements and cannot get use of the extended pros which these features offer. The simple graphic applications which does not require the use of complex three dimensional objects, shading and other advanced graphic features, as is the case of our prototype, can be effectively transformed into pure HTML/CSS and so get all the advantages of not needing any Web Graphical Engine. Problem with pure CSS 3D transformations is connecting elements in space, like arrows and lines. We were able to create such an arrow between elements in 3D space with only CSS 3D transformations, thus still maintaining all the pros which HTML and CSS brings us.

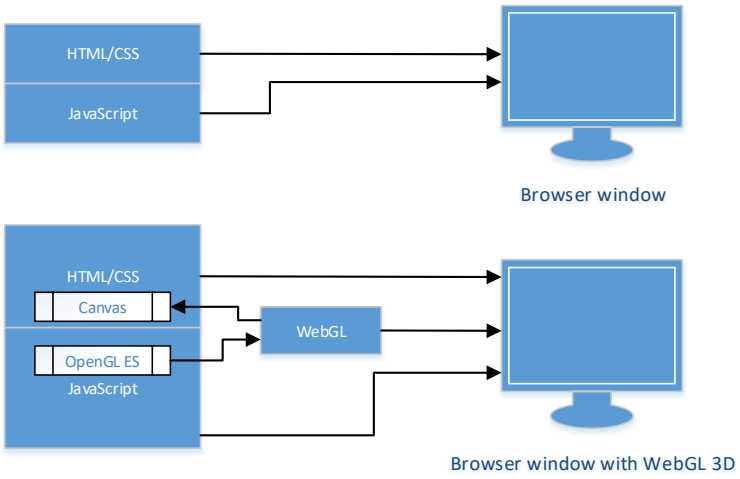


*Figure 3. CSS3D and WebGL architecture for Web App.*

## 4 Conclusion and future work

In migration of graphic desktop applications to web, there is not yet a standardized way of doing it. It is very important to well measure the system to be migrated and choose the appropriate technology. Although losing the power of pure HTML/CSS and encountering problems with synching HTML with WebGL, for applications with complicated geometries and a lot of visual effects is the best way to use WebGL engine for their web applications. Dealing with simple shapes and effects it is very good to stay with pure HTML and CSS 3D transformations. The boom wave of WebGL is on decline and there is a lot of work put into CSS 3D transformation libraries and tools and it looks like in future this could expand even more.

Transforming natural language and written text into diagrams is our nearest goal. We plan to extend our prototype for interpreting natural language into three dimensional diagrams. These diagrams we plan to generate automatically using Force-directed graph layout algorithms which allows weighting of the vertices and the edges for less complex and easier to comprehend 3D diagrams and models.

*Acknowledgement:* This contribution is the partial result of the project Research of methods for acquisition, analysis and personalized conveying of information and knowledge, ITMS 26240220039, co-funded by the ERDF.

## References

[1] Koike H.: The role of another spatial dimension in software visualization. In: *ACM Transactions on Information Systems (TOIS)*, (1993)

[2] Koike H. and Chu H-CH.: How does 3-d visualization work in software engineering? In: *ICSE '98 Proceedings of the 20th international conference on Software engineering*, ICSE Proceedings, (1998)

[3] Irani, P., Ware, C.: Diagrams based on structural object perception. In: *AVI '00 Proceedings of the working conference on Advanced visual interfaces*. AVI Proceedings, ACM New York, (2000). pp. 61–67.

[4] Gil, J. and Kent, S.: Three dimensional software modelling. In *ICSE '98 Proceedings of the 20th international conference on Software engineering*, ICSE Proceedings, (1998).

[5] McIntosh, P.: X3D-UML: User-Centred Design. Implementation and Evaluation of 3D UML Using X3D. PhD. Thesis, RMIT University, (2009).

[6] Duske, K.: *A Graphical Editor for the GMF Mapping Model*. (2010). http://gef3d.blogspot.com/2010/01/graphical-editor-for-gmf-mapping-model.html

[7] Pilgrim J. and Duske K.: Gef3d: A framework for two-, two-and-a-half-, and three-dimensional graphical editors. In: *Proceedings of the 4th ACM Symposium on Software Visualization*, SoftVis '08, pages 95–104, New York, NY, USA, 2008. ACM.

[8] Biederman, I.: Recognition-by-components: A theory of human image understanding. Psychological Review, vol. 94, (1987). pp. 115–147.

[9] *OMG Unified Modeling Language (OMG UML)*, Infrastructure, Version 2.4.1, [Online], Available: http://www.omg.org/spec/UML/2.4.1/, Accessed: March 1, 2014

[10] Clark K.: *Creating 3D worlds with HTML and CSS*, http://www.keithclark.co.uk/articles/creating-3d-worlds-with-html-and-css/

[11] Gregorovic L., Polasek I., and Sobota B.: Software model creation with multidimensional UML. In: *International Conference on Research and Practical Issues of Enterprise Information Systems*, The 23rd IFIP World Computer Congress, Daejeon, Korea, (2015)

[12] Gregorovic L. and Polasek I.: Analysis and Design of Object-oriented Software using Multidimensional UML. In: *I-Know 2015. 15th International Conference on Knowledge Technologies and Data-driven Business*, Graz, Austria. ACM. (2015)

[13] Puder A.: Extending desktop applications to the web. In: *Proceedings of the 2004 international symposium on Information and communication technologies* (ISICT '04). Trinity College Dublin, pp. 8 - 13. (2013)

[14] Belan R.: Transformation of UML Combined Fragments from 2D to 3D. In: *Proceedings: of the 11th Student Research Conference in Informatics and Information Technologies Bratislava*, (2014). pp. 245 – 250.