BabiaXR: Virtual Reality software data visualizations for the Web

David Moreno-Lumbreras*
EID @ Universidad Rey Juan Carlos & Bitergia

Jesus M. Gonzalez-Barahona†
Universidad Rey Juan Carlos

Andrea Villaverde‡
Universidad Rey Juan Carlos

ABSTRACT

We present BABIAXR, a tool to analyze, extract and visualize data in an immersive virtual reality web environment, making use of the WebXR and WebGL standards, making VR visualizations more accessible. The approach focuses on the analysis of a specific visualization, called CodeCity, a well known city metaphor for visualizing source code metrics in a 3D environment.

Index Terms: Data visualization, CodeCity, City Metaphor, Software Visualization, Software Evolution, Reverse Engineering, Virtual Reality, Web, 3D

1 INTRODUCTION

BABIAXR aims to explore the field of data analytics on software development immersed in virtual reality environments. Therefore, it is placed at the confluence of several areas of research and industrial interest: data analytics, analysis of software development processes, and immersive virtual reality environments. These three fields have been developing largely independently, but in recent years they have shown a high degree of convergence. Both the research results and the signs of industrial interest and technological advances make it possible to ensure that this field will be very active in the coming years.

The main objective of this approach is to build an immersion system in virtual reality to facilitate the analysis of software development. The system must be complete enough to be able to perform, visualize, and interact with the most common types of analysis today, but also some of the new types of analysis that are being proposed in the scientific literature. This system should work on multiple platforms, taking advantage of the multiplatform capabilities of WebVR and other related standards. The aim of this system is to improve the possibilities of understanding what is happening in a software development project, and of evaluating the results of any measure taken to try to improve it.

Within the source code visualizations, this approach has focused on the analysis, adaptation and improvement of CodeCity. *Code city* is a well known metaphor for visualizing source code metrics in a 3D environment. First used by Knight and Munro in 1999 [10], it became popular with CoodeCity [23], to date the most impactful tool developed to implement this metaphor, also inspiring some other similar approaches (*e.g.,* [2, 18, 20]). CoodeCity shows software systems as cities by mapping metrics of artifacts (*i.e.,* classes, files) to features of buildings (*i.e.,* height, size, color), and placing buildings in locations related to the position of artifacts in the system hierarchy. Those cities can be intuitively explored [22], offering a clear notion of locality, supporting orientation, and making it explicit the underlying structural complexity.

One of the main contributions of the approach is an open source implementation of CodeCity in BABIAXR which runs on any modern browser, including the browser available on VR devices, making VR visualizations more accessible.

2 BABIAXR

BABIAXR[1] is a toolset for 3D data visualization in the browser. BABIAXR is based on A-FRAME,[2] an open web framework to build 3D, augmented reality (*i.e.,* AR), and VR experiences in the browser. A-FRAME extends `HTML` with new entities allowing to build 3D scenes as if they were `HTML` documents, using techniques common to any front-end web developer. A-FRAME is built on top of THREE.JS,[3] which uses the WEBGL API available in all modern browsers.

BABIAXR extends A-FRAME by providing components to create visualizations, simplify data retrieval, and manage data (*e.g.,* data filtering or mapping of fields to visualization features). Scenes built with BABIAXR can be displayed on-screen, or in VR devices, including consumer-grade headsets. Figure 1 shows a sample scene built with BABIAXR. BABIAXR is open source: Its source code is available on GITLAB[4] and it can be installed with NPM.[5]

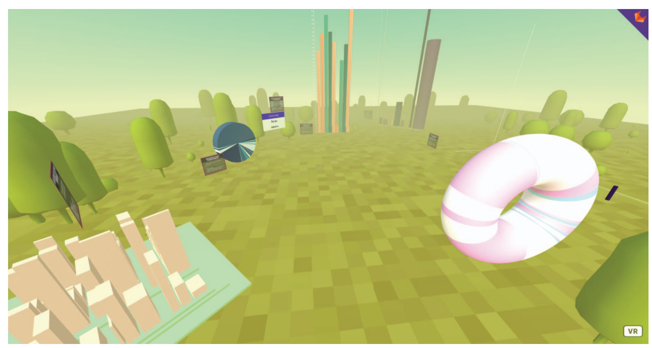

Figure 1: Example of a BABIAXR Scene

2.1 CODECITY-like Visualizations in BABIAXR

BABIAXR provides two visualizations based on CODECITY. We present BABIAXR-CODECITY, which reimplements the original CODECITY but uses a different algorithm to layout the city, and presents the 3D visualization in a web browser (*i.e.,* instead of being a desktop SMALLTALK application). For the city layout, BABIAXR employs a *spiraling algorithm*: the first element is placed at the center of the spiral and the remaining elements spiral around it.

The algorithm is used recursively at all the levels of the software architecture, producing a layout in districts, that are composed of subdistricts, and so on, until the buildings are displayed at the deepest level. Figure 2 shows a scene depicted with BABIAXR-CODECITY.

BABIAXR-CODECITY scenes are interactive. The user can hover the cursor on a building to open a tooltip containing the name of

*e-mail: d.morenolu@alumnos.urjc.es
†e-mail: jesus.gonzalez.barahona@urjc.es
‡e-mail: andrea.villaverde@urjc.es

[1] BABIAXR: **https://babiaxr.gitlab.io**
[2] A-FRAME: **https://aframe.io**
[3] THREE.JS: **https://threejs.org**
[4] **https://gitlab.com/babiaxr/aframe-babia-components**
[5] **https://npmjs.org/package/aframe-babia-components**

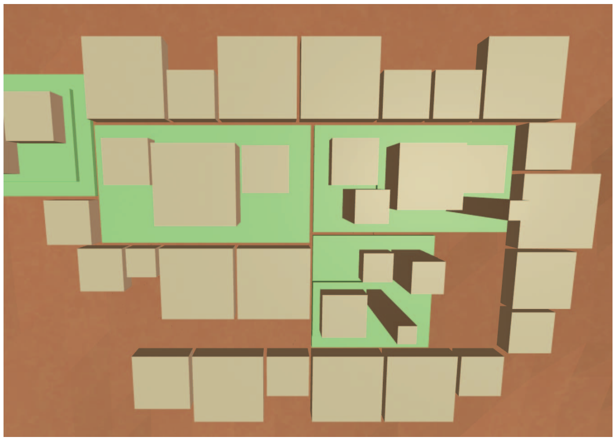

Figure 2: Example of a BABIAXR-CODECITY Scene

the file together with the values of the metrics for the corresponding software artifact (*e.g.,* number of functions, lines of code, and Cyclomatic Complexity Number [9] of a file). The tooltip disappears when the cursor leaves the building, but pinned tooltips can be enabled by clicking on a building. To obtain information on a district (*i.e.,* a folder) the user can click on it. These interactions work in the same way on-screen (*i.e.,* with the mouse as cursor) and in VR (*i.e.,* with the controller of the VR headset as cursor). Like the original CODECITY, BABIAXR-CODECITY maps the values of software metrics to features in the visualization. In the current version of BABIAXR-CODECITY, each building corresponds to a file. Its base area is proportional to the number of functions, its height corresponds to lines of code per function, and its color represents the Cyclomatic Complexity value (*i.e.,* in a blue to red scale).

2.2 From Source Code to a 3D Scene

BABIAXR-CODECITY can produce a scene starting from a commit in a GIT repository (*i.e.,* snapshot). A Python script (*i.e.,* COCOM_GRAAL2ES.PY[6]) clones the repository, checks out a given commit, computes the values of metrics for each file, and stores them in an ELASTICSEARCH[7] database. The script uses GRAAL [5] and PERCEVAL [7] to retrieve and compute the values of metrics. GRAAL, in turn, uses other tools to compute metrics, via its COCOM backend[8]).

Once the results of the analysis are stored in the database, another Python script (*i.e.,* GET_LIST.PY[9]) queries ELASTICSEARCH to produce a JSON document in the format required by BABIAXR-CODECITY that contains all the information needed for the visualization. It is structured as follows:

```
[
  {
    "file_path": "aaa/bbb/ccc"
    "metric": x,
    "metric2": y,
    ...
  },
  ...
]
```

The scene to visualize this data is composed of a single HTML file, which uses the JSON document. The HTML file imports all the dependencies (*i.e.,* A-FRAME and BABIAXR JavaScript packages) and defines the scene by including the corresponding elements and components: `babia-queryjson` to retrieve the JSON document, `babia-treebuilder` to generate the tree-like data structure needed by BABIAXR-CODECITY, and `babia-boats` which is the actual component to generate the visualization. Each component has its own configuration, detailed in the documentation.[10] The listing below shows a sample scene, including some configuration parameters:

```
<a-scene id="scene">
  <a-entity id="rawdata"
    babia-queryjson="url: data.json">
  </a-entity>
  <a-entity id="treedata"
    babia-treebuilder="field: field_list;
    split_by: /; from: rawdata">
  </a-entity>
  <a-entity id="city"
    babia-boats="from: treedata;
    area: metric1; height: metric2; color: metric3">
  </a-entity>
  ...
</a-scene>
```

Figure 3 summarizes the complete workflow to produce a scene with BABIAXR-CODECITY starting from a source code snapshot in a GIT repository.

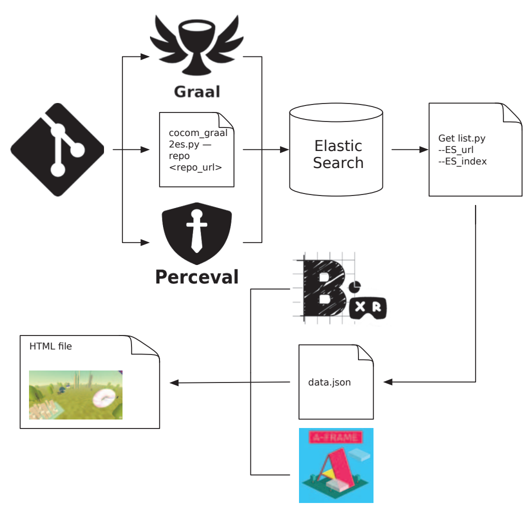

Figure 3: BABIAXR Workflow: From Source Code to a Scene

As we have already said, thanks to modern web technology, these scenes are fully usable in VR mode in any modern browser, including those that include VR goggles, Figure 4 shows an user visualizing data in a VR web scene created with BABIAXR.

Table 1 shows information about the code files developed for this use case, we rely on module reuse, so we are aware of the size and the modules independence, we separate frontend and backend and, in terms of visualizations, each module can be used with others or in standalone mode as the use in this approach. We have tested the size of the city (*i.e.,* number of buildings) that can be shown without

[6] COCOM_GRAAL2ES.PY docs: **https://gitlab.com/babiaxr/aframe-babia-components/-/tree/master/tools/generate_repository_data**

[7] ELASTICSEARCH: **https://www.elastic.co/**

[8] GRAAL-COCOM backend: **https://github.com/chaoss/grimoirelab-graal#backends**

[9] **https://gitlab.com/babiaxr/aframe-babia-components/-/tree/master/tools/generate_from_es**

[10] **https://gitlab.com/babiaxr/aframe-babia-components/-/tree/master/docs/APIs**

Figure 4: Example of a BabiaXR Scene in VR with Oculust Quest 2 glasses

noticeable glitches (at about 30-40 frames per second) in the browser. We have found that an Intel i7 machine with 8 GB of RAM, with an Intel HD620 4K graphics card, can visualize easily cities with about 8,000 buildings both in Firefox and Chrome, on Debian, while the standard browser in Oculus Quest 2 can visualize cities of about 1,000 buildings.

Table 1: Code used basic information

Code	Lines of Code	Size (in KB)	Language
babia-boats.js	1173	43.5	JavaScript
babia-treebuilder.js	298	10.1	JavaScript
babia-queryjson.js	185	4.8	JavaScript
cocom_graal2es.py	100	3.25	Python
get_list.py	227	7.92	Python

In addition, this special visualization of CodeCity is completely integrable with the rest of visualizations and filters, so with BabiaXR complex VR scenes can be created with more than one visualization and/or cities.

3 Premilimary Results

One of the main results obtained with the thesis study is the results obtained in a controlled experiment that compares the versions of CodeCity both in use with Virtual Reality glasses and on a screen. This study was published at VISSOFT 2021 [14]. In summary, we consider that the results of our experiment show that CodeCity in VR is comparable to CodeCity on-screen, with respect to the accuracy of results, and much better with respect to efficiency. When we focus on completion time, the picture is completely different. Despite lacking experience in VR headset usage, BabiaXR VR participants were considerably faster than BabiaXR on-screen participants across all program comprenhension tasks proposed in the experiment. This result suggests that VR immersion could play a pivotal role when carrying out program comprenhension tasks in a 3D environment.

4 Related work

Virtual reality has been shown to facilitate discovery in domains in which space plays an important role. For example in the field of brain tumors [24], perception of shapes and forms [6], paleontology [11], caves [17], and magnetic resonance imaging [4]. Data visualization in virtual reality allows the use of multidimensionality for abstract analysis, and even more so for large data sets.

Regarding the city metaphor implementations, the literature starts with the approach presented by Munro *et al.* [10] for visualizing software systems in a 3D software world, then the literature presents several approaches to support developers on maintaining software systems as fulfilling the program comprehension tasks Wettel *et al.* presented CodeCity [23], one of the most important approaches and a turning point in the literature. Among these studies, there are techniques based on software visualization, and some of them focused on different programming languages.

The integration of VR in this CodeCity visualization is currently active in the literature, Juraj Vincur *et al.* [21], [20] propose a Virtual Reality city for analyzing object-oriented software, made it with non-web technologies, Steinbrückner and Lewerentz [18] propose stable city layouts for evolving software systems, using a different layout than a treemap. Getaviz [1] is another tool that uses the city metaphor in order to generate structural, behavioral, and evolutionary views of software systems for empirical evaluation. In terms of interaction with the city, one good example is CityVR [12] that uses the same metrics as the original CodeCity but adds interactions with a VR headset using the controllers and the sight direction. Fittkau, Krause *et al.* [8] proposed a VR approach for the ExplorViz tool based on the first versions of WebVR, focusing on the runtime and static characteristics of object-oriented programming software systems. Romano *et al.* [3] proposed an approach to visualize Java systems in Unreal Engine 4 using the city metaphor in VR, then they validated the tool with a controlled experiment. Other metaphors have been visited by the literature, Misiak, Schreiber *et al.* [13] proposed the island metaphor for visualizing OSGi-based software systems in VR, introducing and emphasizing in the dependencies visualization.

5 Future Plan

The thesis will continue to focus on the use of CodeCity-type visualizations in Virtual Reality environments, making improvements and different experiments. The current study is based on the temporal evolution of these cities, in order to demonstrate that VR is at least as usable as traditional screens when performing software comprehension tasks when displaying a software project in a city. that evolves over time. Within this specific branch, several studies have been published, Wettel and Lanza [22] proposed a first time evolution CodeCity. *et al.* [19] proposed a different layout for the city, based on streets and sub-streets for the tree structure, allowing to observe the time evolution of the software system. M3tricity [16], a recent re-implementation of CodeCity by the original research group, is web application to visualize software systems as evolving cities that treats evolution as a first-class concept.

In addition, apart from showing source code, another proposed branch of research is the analysis of other metrics performed with software development projects, such as community metrics, activity at the repository level or even the analysis of pull/merge requests and issues. Making use of all the visualizations that are included in BabiaXR. A report has been published in ESEM 2021 for the realization of this study [15]

All this development continues to be in a web environment, making use of WebXR and WebGL for the native implementation of VR.

Acknowledgments

We acknowledge the financial support of the Community of Madrid for the project IND2018/TIC-9669.

References

[1] D. Baum, J. Schilbach, P. Kovacs, U. Eisenecker, and R. Müller. GETAVIZ: Generating structural, behavioral, and evolutionary views of software systems for empirical evaluation. In *2017 IEEE Working Conference on Software Visualization (VISSOFT)*, pp. 114–118, 2017. doi: 10.1109/VISSOFT.2017.12

[2] R. Brito, A. Brito, G. Brito, and M. T. Valente. GoCity: Code city for Go. In *2019 IEEE 26th International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pp. 649–653, 2019.

[3] N. Capece, U. Erra, S. Romano, and G. Scanniello. Visualising a software system as a city through virtual reality. In L. T. De Paolis, P. Bourdot, and A. Mongelli, eds., *Augmented Reality, Virtual Reality, and Computer Graphics*, pp. 319–327. Springer International Publishing, Cham, 2017. doi: 10.1007/978-3-319-60928-7_28

[4] J. J. Chen, H. Cai, A. P. Auchus, and D. H. Laidlaw. Effects of stereo and screen size on the legibility of three-dimensional streamtube visualization. *IEEE Transactions on Visualization and Computer Graphics*, 18:2130–2139, 2012.

[5] V. Cosentino, S. Dueñas, A. Zerouali, G. Robles, and J. M. Gonzalez-Barahona. Graal: The quest for source code knowledge. In *2018 IEEE 18th International Working Conference on Source Code Analysis and Manipulation (SCAM)*, pp. 123–128, 2018. doi: 10.1109/SCAM.2018.00021

[6] C. Demiralp, C. Jackson, D. Karelitz, S. Zhang, and D. Laidlaw. Cave and fishtank virtual-reality displays: A qualitative and quantitative comparison. *IEEE transactions on visualization and computer graphics*, 12:323–30, 05 2006. doi: 10.1109/TVCG.2006.42

[7] S. Dueñas, V. Cosentino, G. Robles, and J. M. Gonzalez-Barahona. Perceval: Software project data at your will. In *Proceedings of the 40th International Conference on Software Engineering: Companion Proceeedings*, ICSE '18, p. 1–4. Association for Computing Machinery, New York, NY, USA, 2018. doi: 10.1145/3183440.3183475

[8] F. Fittkau, A. Krause, and W. Hasselbring. Exploring software cities in virtual reality. In *2015 IEEE 3rd Working Conference on Software Visualization (VISSOFT)*, pp. 130–134, 2015. doi: 10.1109/VISSOFT.2015.7332423

[9] Harrison, Magel, Kluczny, and DeKock. Applying software complexity metrics to program maintenance. *Computer*, 15(9):65–79, 1982. doi: 10.1109/MC.1982.1654138

[10] C. Knight and M. Munro. Comprehension with[in] virtual environment visualisations. In *Proceedings Seventh International Workshop on Program Comprehension*, pp. 4–11, 1999. doi: 10.1109/WPC.1999.777733

[11] B. Laha, D. Bowman, and J. Socha. Effects of vr system fidelity on analyzing isosurface visualization of volume datasets. *IEEE transactions on visualization and computer graphics*, 20:513–22, 04 2014. doi: 10.1109/TVCG.2014.20

[12] L. Merino, M. Ghafari, C. Anslow, and O. Nierstrasz. CityVR: Gameful software visualization. In *2017 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pp. 633–637, 2017. doi: 10.1109/ICSME.2017.70

[13] M. Misiak, A. Schreiber, A. Fuhrmann, S. Zur, D. Seider, and L. Nafeie. IslandViz: A tool for visualizing modular software systems in virtual reality. In *2018 IEEE Working Conference on Software Visualization (VISSOFT)*, pp. 112–116, 2018. doi: 10.1109/VISSOFT.2018.00020

[14] D. Moreno-Lumbreras, R. Minelli, A. Villaverde, J. M. Gonzalez-Barahona, and M. Lanza. CodeCity: On-screen or in virtual reality? In *Working Conference on Software Visualization, VISSOFT 2021, Luxembourg, September 27-28, 2021*, pp. 12–22. IEEE, 2021. doi: 10.1109/VISSOFT52517.2021.00011

[15] D. Moreno-Lumbreras, G. Robles, D. Izquierdo-Cortazar, and J. M. González-Barahona. To VR or not to VR: is virtual reality suitable to understand software development metrics? *CoRR*, abs/2109.13768, 2021.

[16] F. Pfahler, R. Minelli, C. Nagy, and M. Lanza. Visualizing evolving software cities. In *2020 Working Conference on Software Visualization (VISSOFT)*, pp. 22–26, 2020. doi: 10.1109/VISSOFT51673.2020.00007

[17] E. D. Ragan, R. Kopper, P. Schuchardt, and D. A. Bowman. Studying the effects of stereo, head tracking, and field of regard on a small-scale spatial judgment task. *IEEE Transactions on Visualization and Computer Graphics*, 19(5):886–896, 2013. doi: 10.1109/TVCG.2012.163

[18] F. Steinbrückner and C. Lewerentz. Representing development history in software cities. In *Proceedings of the 5th International Symposium on Software Visualization*, SOFTVIS '10, p. 193–202. Association for Computing Machinery, New York, NY, USA, 2010. doi: 10.1145/1879211.1879239

[19] F. Steinbrückner and C. Lewerentz. Representing development history in software cities. In *Proceedings of the 5th International Symposium on Software Visualization*, SOFTVIS '10, p. 193–202. Association for Computing Machinery, New York, NY, USA, 2010. doi: 10.1145/1879211.1879239

[20] J. Vincur, P. Navrat, and I. Polasek. VR City: Software analysis in virtual reality environment. In *2017 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C)*, pp. 509–516, 2017. doi: 10.1109/QRS-C.2017.88

[21] J. Vincur, I. Polasek, and P. Navrat. Searching and exploring software repositories in virtual reality. In *Proceedings of the 23rd ACM Symposium on Virtual Reality Software and Technology*, VRST '17. Association for Computing Machinery, New York, NY, USA, 2017. doi: 10.1145/3139131.3141209

[22] R. Wettel and M. Lanza. Program comprehension through software habitability. In *15th IEEE International Conference on Program Comprehension (ICPC '07)*, pp. 231–240, 2007. doi: 10.1109/ICPC.2007.30

[23] R. Wettel and M. Lanza. Visualizing software systems as cities. In *2007 4th IEEE International Workshop on Visualizing Software for Understanding and Analysis*, pp. 92–99, 2007. doi: 10.1109/VISSOF.2007.4290706

[24] S. Zhang, C. Demiralp, D. Keefe, M. DaSilva, D. Laidlaw, B. Greenberg, P. Basser, C. Pierpaoli, E. Chiocca, and T. Deisboeck. An immersive virtual environment for dt-mri volume visualization applications: a case study. In *Proceedings Visualization, 2001. VIS '01.*, pp. 437–584, 2001. doi: 10.1109/VISUAL.2001.964545