Rethinking Codecity

David Moreno-Lumbreras[1] and Jesus M. Gonzalez-Barahona[2]

[1] Universidad Rey Juan Carlos & Bitergia
dmoreno@bitergia.com
[2] Universidad Rey Juan Carlos
jgb@gsyc.urjc.es

Abstract. Several years ago, a new and different visualization technique of software systems was proposed, the name of this approach is Codecity, and according to the main project page [1]: *Codecity is an integrated environment for OOB software analysis, in which software systems are visualized as interactive, navigable 3D cities.* It relies on the city metaphor to structure the visualization, because it offers a clear notion of locality, thus supporting orientation, and features a structural complexity that cannot be oversimplified. Specifically, Codecity main articles express their layout by the following citation: *The classes are represented as buildings in the city, while the packages are depicted as the districts in which the buildings reside*
In this presentation, I propose the adaptation, reconsideration, and improvement this approach using the current technologies related to the 3D, Virtual Reality (VR) and Extended Reality (XR) inside the web environment. We will also show the analysis of the other and newer implementations of Codecity, analyzing their drawbacks and their advantages according to the first version.
In our case, we rely on A-Frame [2], a web framework for building 3D and VR scenes for the web, for the visualization render engine, developing, thus a new version of Codecity that works in any device that has a web browser and the needed standards included (the most important, WebVR).

Keywords: Software visualization · Web development · Virtual and Extended Reality.

1 Introduction

1.1 Codecity

Codecity [3] is an approach of a 3D visualization which creates cities that look real, due to the combination of layouts, topologies, metric mappings applied at an appropriate level of granularity. It depicts object-oriented software systems as habitable[4] cities that one can intuitively explore. Codecity settles on the city metaphor because it offers a clear notion of locality, thus supporting orientation, and features a structural complexity that cannot be oversimplified. Codecity represents classes as buildings located in districts representing the packages where the classes are defined. The metrics that Codecity uses are the number of methods (NOM) mapped on the building's height and the number of attributes (NOA) on their base size, and for packages the nesting level mapped on the district's color saturation:

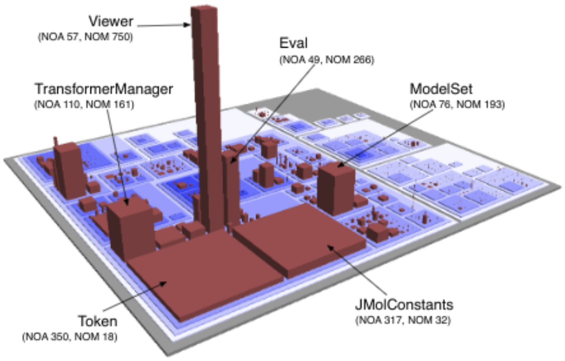

Fig. 1. Example of the Jmol Java package city analyzed with Codecity

One of the known limitations of this approach is that Codecity is strongly bonded to analyze the program structure. We argue that the city metaphor can be more than just analyze code, like analyze the contributions, users, and other kind of metrics related to the community. Other of its limitations is that Codecity doesn't follow a fixed layout for its buildings/districts, it means that every time a code city is generated, the layout and the position of a package could change. This would be an important drawback in terms of time evolution analysis, because the city would change every single time that it's generated.

On the other hand, in terms of interaction, we argue that this visualization type can get more value and better comprehension adding it a Virtual Reality environment, allowing the user to tour and navigate into the city.

2 Current Research

Understanding all the limitations and the possible improvements of the tool, we are going to follow the next steps in our research:

- Reproduce Codecity with its original metrics.
- Reproduce other metrics that have been proposed by other researchers.
- Explore other kind of metrics in the city metaphor.
- Extend the city metaphor, fixed layout, time evolution, etc.

The next points show what we have been doing so far:

2.1 Reproducing Codecity

We are currently working on adapt the initial Codecity visualization on A-Frame. A-Frame [2] is a web framework for building 3D and VR experiences. It is based on top of webVR, an standard for Virtual Reality in the browser. It standard means that this chosen technology works on any browser, so the result of the approach can be used and tested from a computer to a VR headset including smart phones.

First of all, we should think the algorithm that organize the layout of the areas where the buildings are located. We developed a component on A-Frame that just shows the city, using a JSON with an specific format where the city components are defined. In order to generate this JSON with the city structure, another script, in this case using Python, was developed. The idea of this script is to retrieve the data, parse it and enrich it in order to generate the JSON that the A-Frame city component needs. Now, the development is in the status of improvement the algorithm of the layout, having so far the demonstration[3] in the Figure 2.

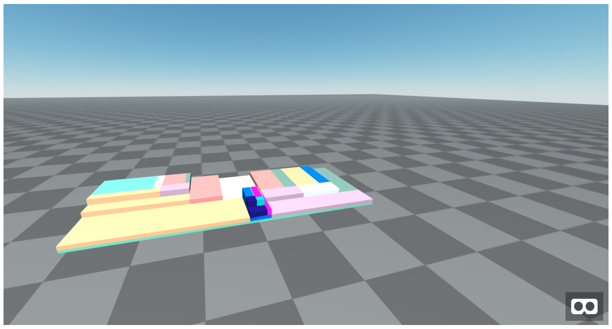

Fig. 2. Current status of the algorithm

2.2 Future work

Once the visualization is totally developed , next step is to reproduce another kind of metrics than other researchers tried to show in the city metaphor, one example is Getaviz [5] that uses the city metaphor in order to generate structural, behavioral, and evolutionary views of software systems for empirical evaluation. Another example is ExplorViz [6] that uses the city metaphor to provide live trace visualization of the communication of the different modules of the program. Finally, in terms of interaction with the city, one good example is CityVR [7] that uses the same metrics than the original Codecity but adds interactions with a VR headset using the controllers and the sight direction.

[3] https://dlumbrer.github.io/aframe-babia-components/examples/test_areas/

All the metrics seen were metrics related to modules or packages. Thus, the next step is to look for the right way to represent different kind of data further than OOB software programs structure, using different kind of metrics. Specifically, we are going to deep in other software development metrics. For instance, the distribution of the commits/repositories, the relationships between the people, time evolution of the code/community, etc. Therefore, we have to face the challenge of move this different kind of data to a city, thinking about what will represent the building, the districts, etc. Making the representation easy to understand and easy to answer questions that the common visualizations don't do.

In terms of the visualization itself, one of the limitations that we should face, is the random layout that Codecity produces. We need to maintain a fixed layout in order to learn and get used to the city of the program analyzed, just so, we can recognize in the future which city belongs to which program/community, identifying correctly the parts of it. This point is very important because of the analysis of time evolution, the idea is to achieve a city that evolves over time, seeing which building/district changes in the same place that it was built.

On the other hand, we are going to research further than the city metaphor, thinking on new ways to visualize this kind of data, applying new ways of layout, like boats in the sea, future cities that are floating in the sky and so on.

We could say that one of the goals is to answer the question *"How software can be represented in Virtual Reality?"*. And including questions like *"Can this be an entry point for a Virtual Reality IDE?"*, and *"Can a stakeholder/ manager/ developer learn the city of its product/ program, and can they analyze it with the city?"*.

References

1. Codecity Web page, https://wettel.github.io/codecity.html (last visited on May 31, 2019)
2. A-Frame.io Web page, https://aframe.io/ (last visited on May 31, 2019)
3. R. Wettel and M. Lanza. Visualizing software systems as cities. In *Proceedings of VISSOFT 2007 (4th IEEE International Workshop on Visualizing Software For Understanding and Analysis)* , pages 9299, 2007
4. R. Wettel and M. Lanza. Program comprehension through software habitability. In *Proceedings of 15th International Conference on Program Comprehension (ICPC 2007). IEEE Computer Society*, 2007.
5. Baum, D.; Schilbach, J.; Kovacs, P.; Eisenecker, U.; Mller, R. Getaviz: Generating Structural, Behavioral, and Evolutionary Views of Software Systems for Empirical Evaluation. *2017 IEEE Working Conference on Software Visualization (VISSOFT)*, 2017.
6. Florian Fittkau, Alexander Krause, Wilhelm Hasselbring (2017): Software landscape and application visualization for system comprehension with ExplorViz. *Information and Software Technology*, Volume 87. pp. 259-277.
7. Merino L Ghafari M Anslow C Nierstrasz. CityVR: Gameful Software Visualization. *International Conference on Software Maintenance and Evolution (ICSME)*, 2017