CityVR: Gameful Software Visualization

Leonel Merino*, Mohammad Ghafari*, Craig Anslow†, Oscar Nierstrasz*
*Software Composition Group, University of Bern, Switzerland
†School of Engineering and Computer Science, Victoria University of Wellington, New Zealand

***Abstract*—Gamification of software engineering tasks improve developer engagement, but has been limited to mechanisms such as points and badges. We believe that a tool that provides developers an interface analogous to computer games can represent the gamification of software engineering tasks more effectively via software visualization. We introduce *CityVR* — an interactive software visualization tool that implements the city metaphor technique using virtual reality in an immersive 3D environment medium to boost developer engagement in software comprehension tasks. We evaluated our tool with a case study based on *ArgoUML*. We measured engagement in terms of feelings, interaction, and time perception. We report on how our design choices relate to developer engagement. We found that developers *i*) felt *curious*, *immersed*, *in control*, *excited*, and *challenged*, *ii*) spent considerable interaction time navigating and selecting elements, and *iii*) perceived that time passed faster than in reality, and therefore were willing to spend more time using the tool to solve software engineering tasks.**
https://youtu.be/R0C-HMAtgnk

I. Introduction

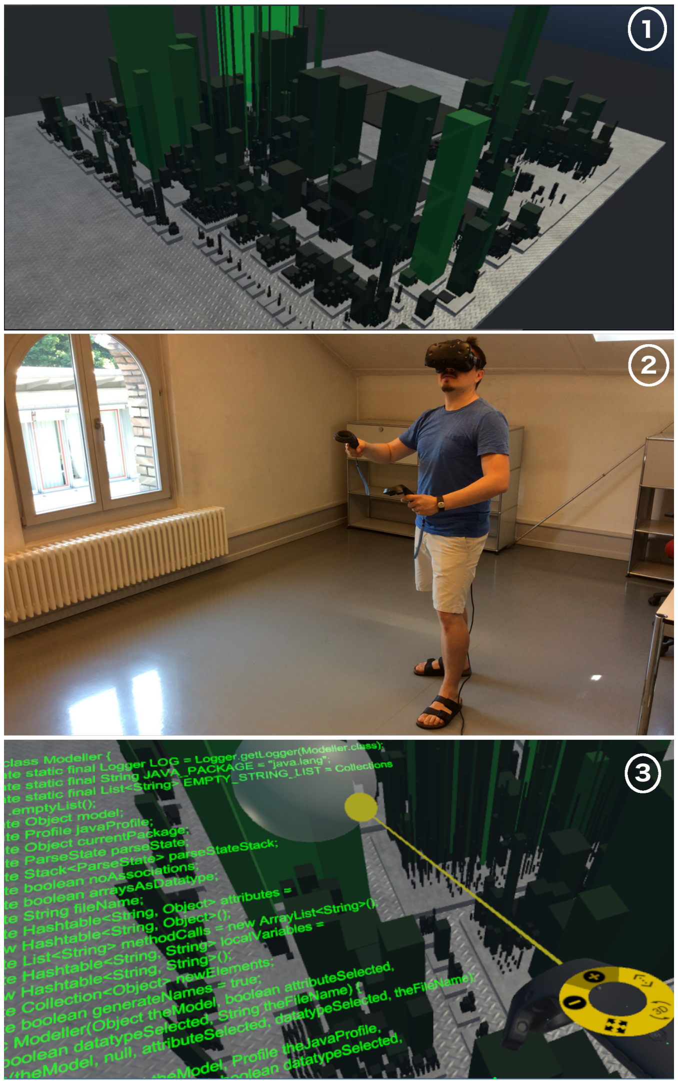
Gamification of software engineering tasks (*i.e.*, applying computer game elements and design techniques) improve developer engagement [1]. Most approaches in software engineering; however, have struggled when putting the concept into action and applied only simple gamification mechanisms such as points and badges [2].

The three main concepts that promote engagement in computer games are *curiosity*, *challenge* and *fantasy* [3]. We observe that developers commonly associate the first two concepts with software visualizations [4]. The third concept, defined as "an illusory appearance"[1], which is also inherent to visualizations. Therefore, we believe software visualizations can represent the gamification of software engineering tasks more effectively. However, we observe that not all software visualization tools promote developer engagement equally.

The medium, technique and interaction are architectural choices in developing software visualizations that can play a role in enhancing user engagement. Consequently, we formulated the following research question:

RQ: How can architectural design choices in developing software visualization tools enhance developer engagement?

We argue, to maximize engagement we need gameful software visualization (*i.e.*, visualization that provides developers with an interface analogous to computer games). We examine our research question by focusing on software comprehension tasks. We designed *CityVR* —an interactive visualization tool that implements the city metaphor technique using virtual reality in an immersive 3D environment (I3D) medium to boost developer engagement in software comprehension tasks (shown in Figure 1). We investigated the effectiveness of CityVR with a case study based on *ArgoUML*, a UML diagramming framework. We measured



```
class Modeller {
ate static final Logger LOG = Logger.getLogger(Modeller.class);
ate static final String JAVA_PACKAGE = "java.lang";
ate static final List<String> EMPTY_STRING_LIST = Collections
.emptyList();
te Object model;
te Profile javaProfile;
te Object currentPackage;
te ParseState parseState;
te Stack<ParseState> parseStateStack;
te boolean noAssociations;
te boolean arraysAsDatatype;
te String fileName;
te Hashtable<String, Object> attributes =
w Hashtable<String, Object>();
te List<String> methodCalls = new ArrayList<String>();
te Hashtable<String, String> localVariables =
w Hashtable<String, String>();
te Collection<Object> newElements;
te boolean generateNames = true;
 Modeller(Object theModel, boolean attributeSelected,
boolean datatypeSelected, String theFileName) {
(theModel, null, attributeSelected, datatypeSelected, theFileName);
theModel, Profile theJavaProfile,
boolean datatypeSelected,
```

Figure 1. *CityVR* gameful software visualization for comprehension. (1) the city metaphor *technique*; (2) a developer in *immersive 3D environment* (I3D) medium; (3) developers *interact* with elements using a controller and a bubble.

[1]"fantasy | phantasy, n." OED Online. Oxford University Press, March 2017. Web. 6 April 2017.

engagement in terms of experienced feelings, interaction, and time perception. We report how our design choices relate to developer engagement. We found that developers *i*) felt *curious*, *immersed*, *in control*, *excited*, and *challenged*, *ii*) spent considerable interaction time navigating and selecting elements, and *iii*) perceived that time passed faster than in reality, and therefore were willing to spend more time using the tool to solve software engineering tasks.

The main contribution of this paper is the discussion of the architectural design choices and lessons learned from building the tool and observing its use. We also contribute to the reproducibility of our research by making the implementation of CityVR publicly available[2].

II. Related Work

Only a few software visualization tools have used virtual reality for software comprehension tasks. FileVis [5] implements a glyph-based visualization, and Software World [6] uses the city metaphor technique. As oppose to CityVR, both tools use a standard computer screen as the medium to display the visualization (probably due to the limited technology available at the time). Two other studies proposed software visualization using other media. *Imsovision* [7] allows developers to visualize software using the CAVE medium. Recently, Fittkau *et al.* [8] evaluated the visualization of software cities using the Oculus Rift device. We observe that neither existing studies elaborate on the methodological principles that support architectural choices in developing visualizations, nor do they discuss the impact of the design decisions in developer engagement, but limit their analysis to performance.

Little research have proposed gamification of software related tasks based on visualization tools. Two software visualization tools have been proposed for teaching software engineering: *1*) CodeSmellExplorer [9] helps students to recognize code smells by interacting with a 2D graph network visualization displayed in a tabletop. In the tool students are challenged to connect physical cards (each listing a given code smell) to refactorings; *2*) Sort Attack [10] implements a 2D visualization based on a standard computer screen, in which a number of game techniques such as lives, levels and time help students to learn algorithms. CodeMetropolis [11] support developers for comprehension of test suites through an enriched software city visualization, implemented using the Minecraft game engine, that is displayed in a standard computer screen.

Instead, CityVR is designed to boost developer engagement during visualization for software comprehension by considering the impact of architectural design choices such as the selected technique, interaction and medium. As opposed to CodeCity [12] that offers interactions based on a computer screen setup, in CityVR developers interact with visualizations by *i*) moving across the available physical space, and *ii*) selecting classes using controllers held in their hands.

[2]http://scg.unibe.ch/research/cityvr

III. CityVR Overview

CityVR is an interactive software visualization tool that is displayed in an I3D medium. We selected I3D as the medium for our visualization since it promotes engagement [13]. CityVR allows developers to obtain an overview of the software system while they are immersed in it. We use the taxonomy proposed by Merino *et al.* [14] to characterize our visualization tool. We first introduce the three dimensions that relate to the problem domain (*i.e.*, audience, task and data) and then the two that relate to our solution: representation and medium. We split the representation dimension into technique and interaction.

CityVR targets the software maintainer *audience* who has to perform software comprehension *tasks* in order to correct and evolve software systems. The *data* available are source code of software systems. We discuss the architectural design choices that we made to boost developer engagement in comprehension tasks.

A. Design

Medium. The complexity of software comprehension tasks requires developers concentration. Typically, developers boost their concentration by isolating themselves from the noise with headphones. We observe that I3D can do the same for their sight. I3D can offer developers a more complete immersion that also includes touch. However, I3D is not popular for software visualization. In previous work we characterized software visualizations and found that most visualizations are designed to be displayed on the standard computer screen [14]. We then studied how other media impact in the effectiveness of software visualizations [4]. We found that developers who visualize systems using an I3D medium exhibit good performance, the best recollection and an adequate user experience (*i.e.*, they feel *fascinated*, *free* and *playful*). Finally, we selected I3D as the medium to display our visualization.

Technique. We selected the city metaphor technique as it has not only proven to be effective to support developers in software comprehension tasks [12], but is also present in popular games such as SimCity, Grand Theft Auto. Figure 1 ① shows a software city. Each building in the city represents a class in the system, the districts represent software packages. The technique can encode three metrics: one in the square base of buildings (*i.e.*, width and depth), one in the height, and one in the color of buildings.

Interaction. We observe that developers spend long hours sitting in a chair or standing in front of a computer. The lack of movement during programming sessions has a negative impact in their daily experience. We conjecture an environment that encourage developers to freely navigate the system (*e.g.*, walk, crouch, jump) without having to stop software comprehension tasks can improve their engagement [15]. In CityVR developers interact with the visualization through navigation and selection. We scaled the visualization of the software system to fit the physical space available in the room where the tool is used. In

CityVR developers can select classes using two controllers with their hands. Figure 1 (3) shows a developer who finds a class of interest, and use one of the controllers to create a *bubble* (pointed by the yellow beam). Then, she drags the bubble with the other controller and drop it in a building to inspect the source code of the represented class. In that way, developers can analyze metric values and inspect source code to get a better understanding of a particular artifact. The source code and metric values are displayed in a panel attached to one of the controllers, so developers can move it with their hand. Developers can also scroll through the code using the buttons placed in one of the controllers (+ and - buttons).

B. Workflow

The four step workflow that developers have to follow when using CityVR are: *1*) *Source code*: a developer creates a model of the system by passing source code files (*i.e.*, Java and C/C++) as input to an MSE exporter (*e.g.*, VerveinJ, inFAMIX, jdt2famix). The output model (.mse file) contains the main characteristics of the system such as inheritance, dependency, and metrics that can be used for analysis. *2*) *System model*: using CodeCity[3] for the Moose 5[4] platform, developers configure a city visualization by defining the mapping between a set of system properties to the dimensions available in the visualization technique. Once developers are happy with the visualization displayed on-screen, they can export the model of the visualization (.csv file). *3*) *Visualization model*: the model contains the selected properties of the system to be visualized and the layout of the buildings of the city. Developers use the visualization model (*i.e.*, .csv file) as input to CityVR in Unity3D 5.5[5]. They can adjust parameters such as the size of the room and compile the application (.exe file). Finally, *4*) *Immersive 3D Software City*: developers can use the visualized system to solve their tasks. We used an HTC Vive VR headset with a 2160 x 1200 combined resolution, 90 Hz refresh rate and 110° field of view. We selected the HTC Vive since it includes the highest number of sensors (among similar devices). Since our interest is to analyze user engagement, we considers these sensors useful.

More implementation details available in CityVR site[2].

IV. Case Study

We investigated the effectiveness of CityVR based on a case study. We configured CityVR to visualize the ArgoUML v.0.34[6] system (as shown in Figure 1). In the software city three metrics are encoded in the properties of buildings, namely the *number of lines of code* (NLOC), the *number of methods* (NOM), and the *number of attributes* (NOA), which are mapped to their color (using a linear transformation), height, and width/depth, respectively. We invited six participants to explore CityVR, and we subsequently conducted semi-structured interviews. They were not paid and freely opted to participate in the study. All of them were experienced developers (*i.e.*, 6.5 ± 1.5 years), and all have an academic background in computer science (*i.e.*, one bachelor, four PhD, and one post-doc). We selected participants with some experience using software visualizations (their self-reported experience ranged between 2 and 5 in a 5-step Likert scale). Participants neither had prior experience using I3D, nor did they have knowledge of the implementation details of ArgoUML. After explaining the encoding used in the visualization and the interactions available, we asked participants to complete two comprehension tasks. The tasks are as follows:

T.1) *How well is ArgoUML designed (*e.g.*, patterns/smells)?*
T.2) *What is the semantics of each package?*

Participants found several code smells such as a bright and massive *god class*, several thin and long *facade* classes, and a few large and flat *data* classes. The code inspection also revealed some of the semantics of packages hidden in the source code. For example, a *configuration* package that contains three data classes with parameters required by various components of the system. Participants also identified a package that contains the implementation of the graphical interface of the system.

These rather difficult tasks were not designed to evaluate the effectiveness of the software city technique but to *stress* navigation and interaction, thus allowing an evaluation of the engagement of participants. We measured engagement in terms of *i*) interaction (*i.e.*, movement), *ii*) feelings, and *iii*) time perception.

Interaction We observed that the more participants engaged, the more they interacted. We analyze participants' engagement by measuring their movement across the physical space. We instrumented our tool to record the position of participants during the visualization of the system. The results of each participant is shown in a separate chart in Figure 2. The marks in the chart represent the position of participants. We observe that participants do feel oriented using the visualization and adopt various strategies to navigate the city *i*) *Center view*. Some participants *[P2]* and *[P6]* opted by exploring the city starting from its center. *ii*) *Diagonal view*. One participant *[P1]* preferred to stand in the empty corners of the city to obtain an overview. *iii*) *Omnidirectional view*. Most participants (*i.e.*, *[P3]*, *[P4]*, and *[P5]*) felt free to explore the particularities of the city by using most of the space available in the physical room. Certainly, navigation alone is not a measure of engagement by itself. Users could move for other reasons without engaging in the activity. However, we believe the combination of objective measures such as navigation with subjective ones such as feelings and time perception do exposes their engagement.

Feelings When participants were using the visualized system, we asked them to share their feelings. Participants found it "nice to walk" across the system, and felt that was

[3]http://smalltalkhub.com/#!/~RichardWettel/CodeCity
[4]http://www.moosetechnology.org/
[5]https://unity3d.com/
[6]https://sourceforge.net/projects/argouml/

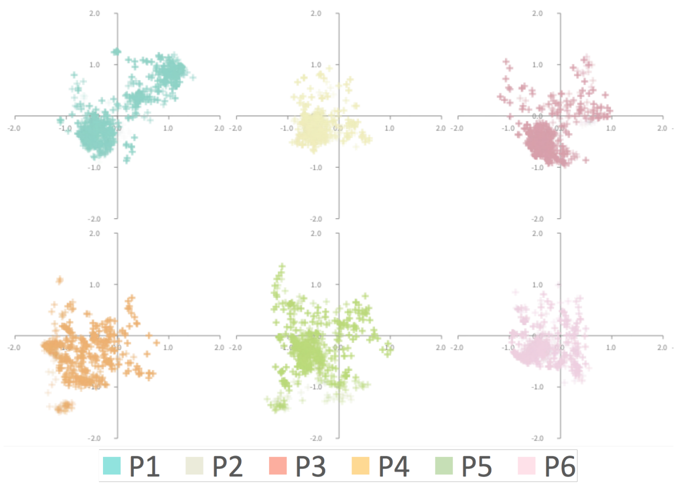


Figure 2. Scatterplots that map the location of participants as they move across the physical room during the visualization of ArgoUML.

fun to interact with the system using their arms and their whole body. We also asked participants to identify their feelings when they finished the tasks. We asked them to select the top five strongest feelings from a list of twenty words (proposed to describe gaming experiences [16]). Frequent feelings were *curious*, *immersed*, *in control*, *excited*, and *challenged*.

Time perception The subjective perception of the passage of time changes according to the engagement of users [17]. When users engage with a task, they tend to lose track of time [18]. Therefore, at various moments of the interview, we asked participants to report how much time they perceived had passed. We asked the first two participants to estimate the time when 10 minutes had passed, and both were correct. We noticed that the time was too little and that participants tend to answer rounded numbers. Therefore, we decided to ask the next four participants to estimate the time when 42 minutes had passed. Three of them perceived that 30 minutes had passed, while one participant was much closer and estimated that 40 minutes had passed. We observe that even though participants moderately underestimated the passage of time, they *felt* that time passed quickly. One participant said that "time had flown very fast"

V. Discussion

We revisit our research question to discuss the effectiveness of decisions made during the design of CityVR. Some decisions such as the selected technique, medium and navigation seem very effective. However, others such as selection and inspection produced mixed results. It seems that interaction is the architectural choice that offers the most room for improving engagement.

Effective choices: medium, technique, and navigation. We observe that the city metaphor technique fits well to I3D. By scaling the software city visualization to the physically available space, developers can navigate the system by walking, which eases navigation (compared to the traditional navigation in computer screens that uses mouse and keyboard). Participants required little training before they felt comfortable with the type of navigation and medium, and were excited to use I3D.

Limited effect choices: code inspection. The panel attached to the right controller served to inspect the source code of a selected class. Developers were able to scroll through the file by waving their arm and by pushing two buttons placed in the controller. Although most participants described this interaction using terms such as *"appealing"*, *"futuristic"*, and *"novel"*, one participant felt that the code was hard to read because it depended on his ability to maintain his arm steady. That participant suggested that having a large fixed panel would ease code reading. Other participants who felt fatigued, noticed high latency of the view when inspecting large source code. Participants seemed happy to interact with source code freely in the 3D space. A participant said *"this is the first time that I actually see the whole code of a class that large"* That participant also observed that seeing the whole code of classes made it easier to understand when a class has too many lines of code and needs refactoring.

Ineffective choices. In CityVR developers select classes for inspection using a bubble, which they create and drag to buildings to obtain details-on-demand of represented classes. However, we observed that developers found it difficult to use. Sometimes developers forgot the mechanism to create and drag the bubble. Other times they lost the bubble inside buildings and had to create a new one.

We observe that extending CityVR to other tasks (*e.g.*, testing, debugging) would require to mitigate the cost of forcing developers to leave their IDE. One solution to that problem would be to transfer the whole IDE to the I3D medium. We also observe that even though the isolating effect of I3D can help developers to concentrate, it could also introduce a social debt. We think that collaborative visualization could mitigate that effect. We envision developers in remote locations using virtual reality as well as co-located developers using augmented reality for visualizing systems collaboratively.

VI. Conclusion and Future Work

We introduced *CityVR* —a tool that implements the software city technique using an immersive 3D environment medium. Through a case study we analyze how developers engage with the tool. We found that developers *i*) felt *curious*, *immersed*, *in control*, *excited*, and *challenged*, *ii*) spent considerable interaction time navigating and selecting elements, and *iii*) perceived that time passed faster than in reality, and therefore were willing to spend more time using the tool to solve software comprehension tasks. In the future we plan to expand this work by *i*) exploring other visualization techniques, interactions and media, and *ii*) investigating how I3D can support remote collaborative software visualization.

References

[1] D. J. Dubois and G. Tamburrelli, "Understanding gamification mechanisms for software development," in *In Proc. of FSE.* ACM, 2013, pp. 659–662.

[2] O. Pedreira, F. García, N. Brisaboa, and M. Piattini, "Gamification in software engineering–a systematic mapping," *Journal of the American Society for Information Science and Technology*, vol. 57, pp. 157–168, 2015.

[3] T. W. Malone, "What makes things fun to learn? heuristics for designing instructional computer games," in *In Proc.of SIGSMALL.* ACM, 1980, pp. 162–169.

[4] L. Merino, J. Fuchs, M. Blumenschein, C. Anslow, M. Ghafari, O. Nierstrasz, M. Behrisch, and D. Keim, "On the impact of the medium in the effectiveness of 3D software visualization," in *In Proc. of VISSOFT.* IEEE, 2017, to appear.

[5] P. Young and M. Munro, "Visualising software in virtual reality," in *In Proc. of IWPC.* IEEE, 1998, pp. 19–26.

[6] C. Knight and M. Munro, "Comprehension with [in] virtual environment visualisations," in *In Proc. of IWPC.* IEEE, 1999, pp. 4–11.

[7] J. I. Maletic, J. Leigh, A. Marcus, and G. Dunlap, "Visualizing object-oriented software in virtual reality," in *In Proc. of IWPC.* IEEE, 2001, pp. 26–35.

[8] F. Fittkau, A. Krause, and W. Hasselbring, "Exploring software cities in virtual reality," in *In Proc. of VISSOFT.* IEEE, 2015, pp. 130–134.

[9] F. Raab, "CodeSmellExplorer: Tangible exploration of code smells and refactorings," in *In Proc. of VL/HCC.* IEEE, 2012, pp. 261–262.

[10] A. Yohannis and Y. Prabowo, "Sort attack: Visualization and gamification of sorting algorithm learning," in *In Proc. of VS-Games.* IEEE, 2015, pp. 1–8.

[11] G. Balogh, T. Gergely, A. Beszédes, and T. Gyimóthy, "Using the city metaphor for visualizing test-related metrics," in *In Proc. of SANER*, vol. 2. IEEE, 2016, pp. 17–20.

[12] R. Wettel, M. Lanza, and R. Robbes, "Software systems as cities: a controlled experiment," in *In Proc. of ICSE.* ACM, 2011, pp. 551–560.

[13] E. Brown and P. Cairns, "A grounded investigation of game immersion," in *In Proc. of CHI.* ACM, 2004, pp. 1297–1300.

[14] L. Merino, M. Ghafari, and O. Nierstrasz, "Towards actionable visualisation in software development," in *Proc. of VISSOFT.* IEEE, 2016.

[15] C. Anslow, S. Marshall, J. Noble, E. Tempero, and R. Biddle, "User evaluation of polymetric views using a large visualization wall," in *In Proc. of SOFTVIS.* ACM, 2010, pp. 25–34.

[16] J. Gow, P. Cairns, S. Colton, P. Miller, and R. Baumgarten, "Capturing player experience with post-game commentaries," in *In Proc. of CGAT*, 2010.

[17] D. Baldauf, E. Burgard, and M. Wittmann, "Time perception as a workload measure in simulated car driving," *Journal of Applied Ergonomics*, vol. 40, no. 5, pp. 929–935, 2009.

[18] C. Jennett, A. L. Cox, P. Cairns, S. Dhoparee, A. Epps, T. Tijs, and A. Walton, "Measuring and defining the experience of immersion in games," *International Journal of Human-Computer Studies*, vol. 66, no. 9, pp. 641–661, 2008.