Noname manuscript No.
(will be inserted by the editor)

The City Metaphor in Software Visualization: Feelings, Emotions, and Thinking

Simone Romano · Nicola Capece · Ugo Erra · Giuseppe Scanniello · Michele Lanza

Received: date / Accepted: date

Abstract Software visualization is a program comprehension technique used in the context of software maintenance, reverse engineering, and software evolution analysis. In the last decade, researchers have been exploring 3D representations for visualizing programs. Among these representations, one of the most popular is the *city metaphor*, which represents a target program as a city. Recently, this metaphor has been also implemented in interactive software visualization tools using Virtual Reality (VR) in an immersive 3D environment medium. We report the results of a study to assess the city metaphor implemented in a VR-based tool and in a 3D-based tool with respect to users' feelings, emotions, and thinking. To this end, we contrasted these tools with a non-visual exploration tool (*i.e.,* Eclipse).

The main result of our study is: the use of the city metaphor implemented in a VR-based tool positively affects users' feelings and emotions, while the thinking about this implementation is positive and comparable with that of a traditional 3D implementation of the city metaphor and it is slightly better than the thinking about a non-visual exploration tool (*i.e.,* Eclipse).

Simone Romano
University of Bari, Italy
E-mail: simone.romano@uniba.it

Nicola Capece
University of Basilicata, Italy
E-mail: nicola.capece@unibas.it

Ugo Erra
University of Basilicata, Italy
E-mail: ugo.erra@unibas.it

Giuseppe Scanniello (corresponding author)
University of Basilicata, Italy
Tel.: +39 0971 205881
Fax: +39 0971 205896
E-mail: giuseppe.scanniello@unibas.it

Michele Lanza
Università della Svizzera italiana (USI), Switzerland
E-mail: michele.lanza@usi.ch

Keywords Code City · Software Visualization · Virtual Reality · Feelings · Emotions · Thinking

1 Introduction

High tech companies, *e.g., Google* and *Facebook*, are well known for how they deal with the work activities of their employees. For example, it is of primary relevance for these companies to have fun while employees work and to eat good food at working places during working hours. The underlying assumption seems to be that happy developers work better than unhappy ones [13].

The software visualization research area has proposed techniques and methods for graphically representing aspects of software [32]. In the last two decades, we have witnessed a proliferation of visualization approaches to support a broad range of software engineering activities such as software maintenance, reverse engineering, and software evolution analysis [16,40]. Researchers have been exploring 3D representations for visualizing programs. Among these representations, one of the most popular is the *city metaphor* [18]. It represents a subject program as a virtual city in which developers can navigate and interact with. Recently, the city metaphor has been implemented in interactive software visualization tools that use Virtual Reality (VR) in an immersive 3D environment medium (*e.g.,* [5,23]).

To assess the benefits derived from the use of software visualization approaches, a few quantitative empirical studies have been conducted so far (*e.g.,* [22,40]). Even more surprisingly, how developers feel while using software visualization tools has not been investigated at all.

In this paper, we present the results of a study to assess the city metaphor with respect to users' feelings, emotions, and thinking. In particular, we studied implementations of the city metaphor available in a 3D environment shown in a standard display and in an immersive VR, and a popular IDE (*i.e.,* Eclipse). We can summarize the most important takeaway results of our study as follows: the use of VR positively affects users feelings and emotions, while the thinking about this implementation is positive and comparable with that of the 3D-based version of the city metaphor and it is slightly better than the thinking about Eclipse.

Paper Structure. In Section 2, we provide background information to better understand the research presented in this paper. In Section 3, we summarize work related to ours. We describe the empirical assessment of our research in Section 4, while the results are highlighted and discussed in Section 5. We conclude the paper in Section 6.

2 Background

In the last decade, researchers have been exploring 3D representations for visualizing programs [16]. Among these 3D representations, one of the most popular is the city metaphor (*e.g.,* [1,2,23,37,40]). This metaphor was initially devised to let developers solve high-level program comprehension tasks on a unique version of a target program [37]. Then it was utilized to analyze the evolution of programs [38] and identify possible design issues [39]. Recently, the city metaphor has been implemented in interactive software visualization tools that use VR in an immersive 3D environment medium (*e.g.,* [5,23]).

We first describe the city metaphor and then our 3D- and VR- based implementations. We conclude this section by outlining the planning of our study.

2.1 City Metaphor

The city metaphor leverages the analogy between programs and cities to represent programs—*e.g.,* buildings depict types (*i.e.,* classes or interfaces) and city districts depict packages. Researches have instantiated the city metaphor differently (*e.g.,* [1,2,23,37,40]). That is, despite these instances depicted programs as cities, there were some differences when comparing an instance with another one. For example, the instance proposed by Bacchelli *et al.* [1] depicts interfaces as "circular" buildings, while the instance by Wettel *et al.* [40] represents interfaces (and classes as well) as "square" buildings. We consider in this paper the instance by Wettel *et al.* [40], which we simply call city metaphor from here on. We do not believe that the use of an instance with respect to another represents a threat to the validity of the results because all these instances present a number of similarities and share the goal of representing a subject program as a city.

In Figure 1, we show how the city metaphor visually represents a program, (*i.e., FindBugs*). In the city metaphor, "square" buildings depict types—both classes and interfaces. That is, buildings are drawn as parallelepipeds with square bases. The base size reflects the Number Of Attributes (NOA) of a given class/interface, while the height reflects Number Of Methods (NOM). Therefore, the larger and taller the building, the higher the number of attributes and methods is, respectively. The color of the building, in a scale from dark blue to light blue, reflects the Lines Of Code (LOC)—the darker the building, the fewer the lines of code are. City districts represent packages. Therefore, buildings in the same city district correspond to types in the same package. The color of the city district gives an indication of how much a package is nested—a dark grey color indicates a high

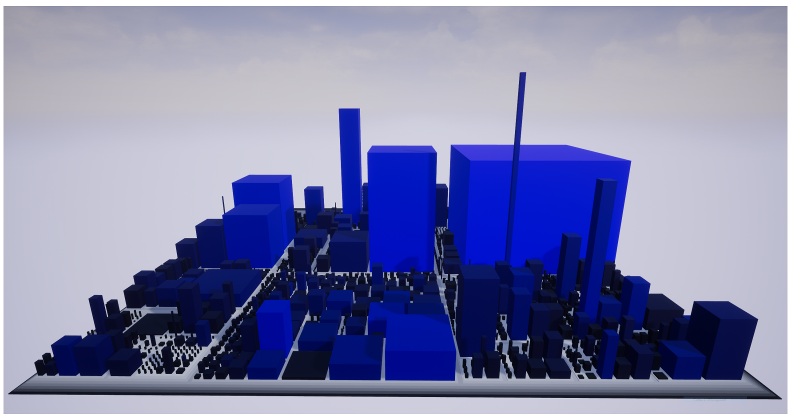
Fig. 1 A program (*i.e.,* FindBugs) represented through the city metaphor.

nesting level for that package (*e.g.,* the root package), while a light grey color indicates a low nesting level.

2.2 Tool Support

In this section, we describe how the city metaphor has been implemented in two tools: the former is based on a traditional 3D visualization while the latter exploits VR. Further details on these implementations can be found in [5].

2.2.1 The 3D-based implementation

Code2City is the tool implementing the city metaphor in a traditional 3D visualization medium. The user interacts with the tool by using mouse and keyboard. These devices allow moving inside the city in any direction. Code2City also offers a number of features to deal with a subject program. The user can identify an object of interest (*i.e.,* building or district) and select it through a viewfinder. When an object is selected, it changes its color (becoming yellow) and further information is shown. If the selected object corresponds to a type, Code2City shows: the name, NOM, NOA, LOC, and belonging package (see Figure 2). If that object corresponds to a package, Code2City shows: its (full) name and the contained classes.

Code2City allows the user to search objects (*e.g.,* types) by name and by callers. The latter allows identifying the types which call the methods of a type provided by the user. Types that respect the search criterion are shown in red.

2.2.2 The VR-based implementation

We named the tool that provides an immersive VR-based implementation of the city metaphor as *Code2City*$_{VR}$. Code2City and Code2City$_{\mathrm{VR}}$ differ only for the medium used to depict the cities. This is to say that they have the same features, while the interaction between the user and the city is different. In particular,

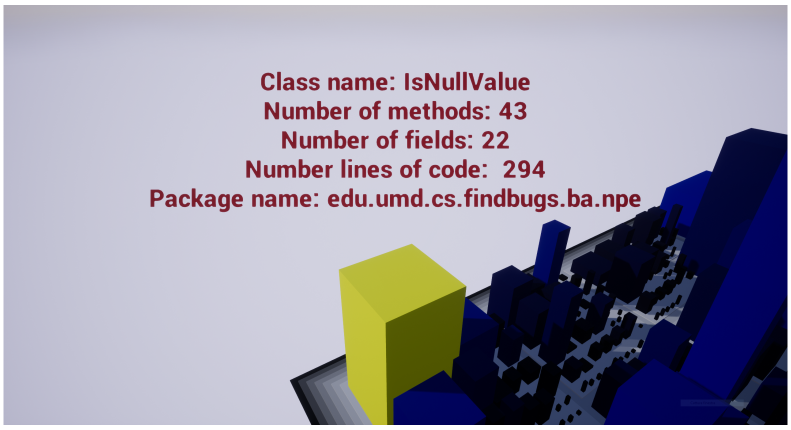
Class name: IsNullValue
Number of methods: 43
Number of fields: 22
Number lines of code: 294
Package name: edu.umd.cs.findbugs.ba.npe

Fig. 2 A type (*i.e.,* `IsNullValue`) highlighted (in yellow) in Code2City. Some information about this type is also displayed.

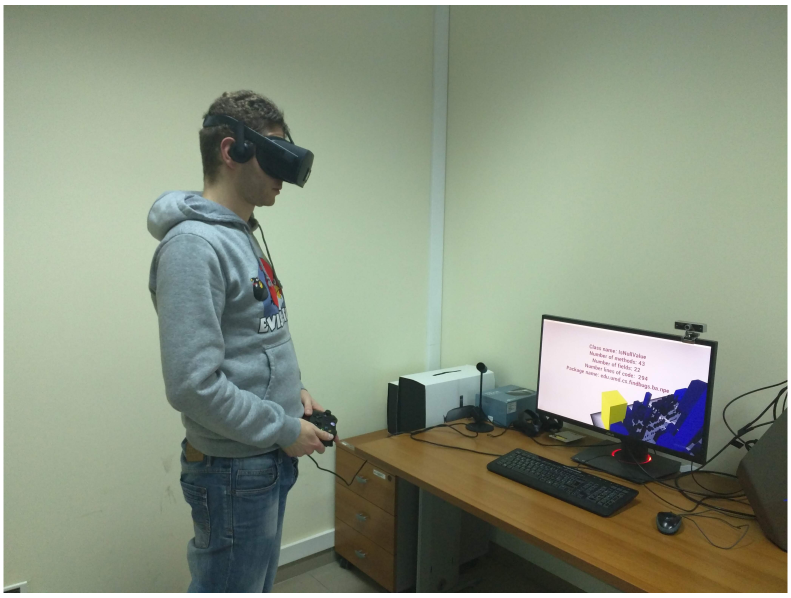

Fig. 3 A user interacting with Code2City$_{\text{VR}}$.

the user exploits a controller and a head-mounted display (*i.e.,* Oculus Rift) to immerse herself in the 3D environment. In Figure 3, we show a user interacting with Code2City$_{\text{VR}}$.

To select an object in Code2City$_{\text{VR}}$, it is sufficient to look at that object and push a controller button. Code2City$_{\text{VR}}$ has been developed so that the size of the scene displayed is proportional to the size of the user so as to provide the feeling of being in a real city. Through VR the user can: *(i)* fly over the city (bird's-eye view); *(ii)* climb the buildings; *(iii)* look at the city from the higher buildings; and *(iv)* walk around the city.

The user can change her movement speed and perform searches. To support searches, a virtual keyboard is visualized. Code2City$_{\text{VR}}$ also provides a text completion feature to reduce the writing time.

2.3 Study Summary

We have planned and conducted a large empirical study with two main perspectives: the former is quantitative, while the latter is qualitative. According to these two perspectives, we gathered different data with the goals of: *(i)* quantitatively measuring the benefits (if any) of using Code2City and Code2City$_{\text{VR}}$ in program comprehension with respect to Eclipse; and *(ii)* qualitatively studying what are the users' feelings, emotions, and thinking while using these tools. This is to say that the research goals related to these two perspectives were different one another, so indicating as the most sound alternative that of presenting quantitative and qualitative results separately. The results of the first part of this large

Table 1 Summary of the study.

Participants	42 Students (34 undergraduates in Computer Science, 8 graduates in Computer Engineering)
Object	FindBugs (1,744 types, 10,123 methods, 4,836 attributes, and 92,571 lines of code)
Design	One factor, three treatments
Factor	Tool (*i.e.,* Code2City vs. Code2City$_{\mathrm{VR}}$ vs. Eclipse)
Tasks	8 comprehension tasks

empirical study is available in the paper by Romano *et al.* [26], while the second part is new and it is presented in this paper.

In the following of this section, we describe those parts of our large empirical study that are common to both investigated perspectives. A summary is reported in Table 1. We asked the participants in the study to execute eight comprehension tasks (see Appendix A). We opted for such a kind of tasks because many software engineering activities deal with program comprehension: software maintenance and evolution, testing, fault localization, quality assurance, reuse, and software integration are only a few examples [4]. In particular, we asked the participants to comprehend FindBugs. The tasks we used in our study were those that Wettel *et al.* conceived for their experiment [40]. This design choice allowed us to mitigate *experimenter expectancies* [41], *i.e.,* the bias experimenters introduce both consciously and unconsciously based on what they expect from the experiment. In addition, the use of FindBugs allowed us to mitigate external validity (*interaction of setting and treatment*) because it was large enough and not obvious to justify the use of software visualization (see Table 1).

We opted for the *one factor with more than two treatments* design [41]. The factor was *Tool*, which assumed three levels: Code2City, Code2City$_{\mathrm{VR}}$, and Eclipse. Each level represents a treatment. The former two treatments were introduced to compare the two different environments showing the city metaphor (*i.e.,* 3D-based vs. VR-based), while Eclipse (*i.e.,* the control treatment) was chosen because it is an IDE widely used in academia and industry. According to the experimental design, each participant received one treatment only. That is, the participant experimenting Code2City used neither Code2City$_{\mathrm{VR}}$ nor Eclipse, and so on. It is worth mentioning that the choice of a within-subject design (*i.e.,* a design where each participant receives any treatment) would have exposed us to the risk of *carryover* [35]. Our design choice allowed us to avoid such a risk because carryover cannot affect a study like ours.

The assignment of the participants to the treatment groups was done randomly. Initially, 54 students accepted to take part in the study, which were randomly divided into the Code2City, Code2City$_{\mathrm{VR}}$, or Eclipse group. Each group consisted of 18 students. Some students, after taking part in the training sessions about Code2City, Code2City$_{\mathrm{VR}}$, or Eclipse, did not take part in the actual experimental session. In particular, 6 and 1 participants assigned to Code2City and Code2City$_{\mathrm{VR}}$, respectively, did not turn up in that session. As for the Eclipse group, 5 participants did not turn up. In the end, 13 participants were administered with Eclipse (two of them were graduates), while 12 (three graduates) and 17 (three graduates) participants were administered with Code2City and Code2City$_{\mathrm{VR}}$, respectively. We could not reallocate participants among the groups because of the training session. That is, each participant had to only took part in the training session on the tool they would use in the actual experimental session.

The participants went through the following procedure:

1. We asked them to fill in a pre-questionnaire to gather their demographic information.
2. We randomly assigned each participant to a treatment group: Code2City, Code2City$_{\mathrm{VR}}$, or Eclipse.
3. The participants in the Code2City group received (by email) a tutorial on the Code2City's features needed to fulfill the program comprehension tasks. Similarly, the participants in the Code2City$_{\mathrm{VR}}$ and Eclipse groups received a tutorial on the tool to be used in the experimental session. We then asked the participants to follow the received tutorial and practice the tool on their own. Successively, these participants were involved in a training session. In this session, we asked the participants to accomplish tasks similar to those of the experimental session, but on a different program—*Jmol* was used in the training session. The goal of the training session was to let participants increase their expertise with the tool they would had used in the experimental session and practice the steps to be followed in that session.
4. We randomly assigned each participant to a computer of a laboratory at the University of Basilicata where they found the tool (*i.e.,* Code2City, and Eclipse) and experimental object (*i.e.,* FindBugs). Thus, we gave the participants the program comprehension tasks. As for Code2City$_{\mathrm{VR}}$, we conducted individual sessions because of the availability of only one head-mounted display.

3 Related Work

In recent years, the video game industry has shown a great interest in VR technologies. The context of use of these technologies has also been extended to other application contexts such as military, educational, and software engineering. For example, Maletic *et al.* [19] described a system called *Imsovision* for the visualization of the object-oriented program. The user wore pair of lightweight liquid crystal shutter glasses for resolving stereoscopic images and used a controller to interact with the scene. Kapec *et al.* [15] presented a visualization system that allows visually analyzing graph structures representing programs in both VR and augmented reality.

The city metaphor has been implemented in a number of VR-based tools [5, 9, 23, 28]. The great interest behind this metaphor is probably to the fact that it has been empirically assessed. In particular, Wettel *et al.* [40] conducted an experiment with the goal of evaluating the use of the city metaphor when performing program comprehension tasks. The authors involved 45 participants from both industry and academia. Each participant was asked to perform comprehension tasks on Java programs (FindBugs or *Azureus*) with the support of *CodeCity* (*i.e.,* the 3D-based implementation of the city metaphor by Wettel *et al.*) or Eclipse. The results suggested that the CodeCity's users attained solutions the to program comprehension tasks that were significantly better as compared with the Eclipse's users (+24%). Moreover, the time to fulfill these tasks was significantly inferior (-12%). This experiment was replicated by the Romano *et al.*'s experiment [26], which extend the validity of the results of the experiment by Wettel *et al.* [40].

Fittkau *et al.* [9] proposed an approach for the exploration of programs, represented as cities, in VR. The authors implemented this approach in *ExplorViz* where the user interacted by means of the Oculus Rift and Microsoft Kinect devices. ExplorViz provided different gestures for the interaction between the user and the visualized city. They then conducted 11 structured interviews to investigate the usability of these gestures. The results indicated that the gestures for translation, rotation, and selection were considered highly usable, while the zooming gesture was not.

Merino *et al.* [23] proposed *CityVR*—an interactive visualization tool that implements the city metaphor by using VR in an immersive 3D environment. CityVR targeted the software maintainer, who has to perform software comprehension tasks to correct and evolve a program. The authors presented the results of an preliminary qualitative empirical evaluation, namely semi-structured interviews with six experienced developers. The most important findings can be summarized as follows: *(i)* developers felt curious, immersed, in control, excited, and challenged when using CityVR; *(ii)* they spent considerable interaction time navigating and selecting elements, and *(iii)* they were willing to spend more time using CityVR to solve software comprehension tasks since they perceived that time passed faster than in reality.

Later, Merino *et al.* [22] conducted a controlled experiment with nine participants and a qualitative study to compare the visualization of program as cities on a standard computer screen with an immersive augmented reality environment in the context of program comprehension tasks. They found that immersive augmented reality facilitates navigation and reduces occlusion, while performance (*i.e.,* completion time, correctness, and recollection) is adequate, and developers obtain an outstanding experience.

Rudel *et al.* [28] proposed *EvoStreets* a kind of software city. The authors conducted a controlled experiment with 20 participants, who were asked to use two variants of EvoStreets: one with head-mounted display (and hand controllers) and a 3D desktop visualization on a standard display (with keyboard and mouse interaction). The main goal of this experiment was to compare these two variants with respect to the navigability (effectively and efficiently) of a scene, namely the visualized software project. An efficient navigation took place when participants spent a low time for homing tasks (finding back to the initial starting point) after navigating the scene. A homing task is considered effective when the participant is found back to the initial position. To deepen the main goal of their experiment, the authors also analyzed the effect of participants' experience with videogames. The main outcome was that head-mounted display and hand controllers did not make it easier to gain a spatial orientation inside of software cities. In addition to that, gamers had an advantage in navigation speed (but not in effectiveness), hinting at that interaction skills may have an influence, and overall efficiency may be increased if one learns an interaction well.

Souza *et al.* [30] proposed a visualization of a subject program through augmented reality using the city metaphor to represent the evolution of software. Their system, called *SkyscrapAR*, represents packages such as districts, sub-packages as stacked districts and classes as buildings positioned in their respective packages. The authors did not conduct any empirical assessment of SkyscrapAR.

The adoption of empirical methods to assess approaches and their supporting tools is not so common in the software visualization field (*e.g.,* [10,11,21,22,31, 40]). The things are even worse in the context of the assessment of users' feelings, emotions, and thinking, although there is a growing interest on human factors in the software engineering field (*e.g.,* [12,17,20,24,25,27]). Our investigation has the merit to fill this gap and then advances the state of the art of software visualization: feelings, emotions, and thinking cannot be considered of secondary importance because they are relevant for user's daily working activities.

4 Assessing Feelings, Emotions, and Thinking

We defined and studied the following Research Question (RQ):

RQ. Might the use of 3D- and VR- based environments implementing of the city metaphor affect users' feelings, emotions, and thinking?

To study RQ, we compared the feelings, emotions, and thinking of developers who experimented Code2City and Code2City$_{\mathrm{VR}}$ with the feelings, emotions, and thinking of those who used Eclipse. To this end, we asked the participants to fill in a PANAS (Positive Affect and Negative Affect Schedule) questionnaire [36] and a post-mortem questionnaire after the participants had accomplished the comprehension tasks.

4.1 PANAS Questionnaire

In Figure 4, we report the PANAS questionnaire. It is a self-report questionnaire consisting of twenty items, each corresponding to a feeling/emotion. Thus, a subject is asked to rate (from 1="very slightly or not at all" to 5="extremely") the

Indicate to what extent you feel this way right now:

	Very slightly or not at all (1)	A little bit (2)	Moderately (3)	Quite a bit (4)	Extremely (5)
1. Interested	☐	☐	☐	☐	☐
2. Distressed	☐	☐	☐	☐	☐
3. Excited	☐	☐	☐	☐	☐
4. Upset	☐	☐	☐	☐	☐
5. Strong	☐	☐	☐	☐	☐
6. Guilty	☐	☐	☐	☐	☐
7. Scared	☐	☐	☐	☐	☐
8. Hostile	☐	☐	☐	☐	☐
9. Enthusiastic	☐	☐	☐	☐	☐
10. Proud	☐	☐	☐	☐	☐
11. Irritable	☐	☐	☐	☐	☐
12. Alert	☐	☐	☐	☐	☐
13. Ashamed	☐	☐	☐	☐	☐
14. Inspired	☐	☐	☐	☐	☐
15. Nervous	☐	☐	☐	☐	☐
16. Determined	☐	☐	☐	☐	☐
17. Attentive	☐	☐	☐	☐	☐
18. Jittery	☐	☐	☐	☐	☐
19. Active	☐	☐	☐	☐	☐
20. Afraid	☐	☐	☐	☐	☐

Fig. 4 PANAS questionnaire [36].

extent to which she feels each of these feelings/emotions. The items in the PANAS questionnaire refer to either positive or negative affects—ten items (*i.e.,* 1, 3, 5, 9, 10, 12, 14, 16, 17, and 19) comprise the positive affect scale while the remaining ten items (*i.e.,* 2, 4, 6, 7, 8, 11, 13, 15, 18, and 20) comprise the negative affect scale. These two scales allow measuring positive and negative affects [36]. In particular, positive affects are measured by means of the Positive Affect Score (**PAS**), which is computed by summing the scores in the positive affect scale [36]. Therefore, PAS ranges in between 10 and 50 where high PAS values indicate high levels of positive affects. Thus, the higher the PAS value, the better it is. As for the negative affects, they are measured through the Negative Affect Score (**NAS**), which is computed by summing the scores in the negative affect scale [36]. Similar to PAS, NAS ranges in between 10 and 50 but, in this case, the lower the value, the better it is. This is because low NAS values indicate low levels of negative affects.

Among the instruments to assess positive and negative affects [14], we opted for PANAS because it has been proven to be a reliable and valid instrument for the assessment of positive and negative affects [36].

4.2 Post-Mortem Survey

In Figure 5, we show the post-mortem survey we used to gather the thinking of the participants about Code2City, Code2City$_{\mathrm{VR}}$, or Eclipse with respect to eight themes: *(i) Tool Quality*; *(ii) Information Quality*; *(iii) Service Quality*; *(iv) Playfulness*; *(v) Perceived Ease of Use*; *(vi) Perceived Usefulness*; *(vii) Attitude toward the Use*; and *(viii) Behavioural Intention to Use.* The post-mortem survey was inspired to the one by Ahn *et al.* [33], which comprised a number of questions grouped according to the above-mentioned themes. Each question was formulated as a statement to be rated. That is, a survey respondent had to rate (from 1=“I strongly disagree” to 7=“I strongly agree”) the extent to which she disagreed/agreed with that statement. In addition to the questions shown in Figure 5, we asked the participants to mention if they had some physical discomforts while using Code2City$_{\mathrm{VR}}$.

To have an overall measure that quantified the participants’ thinking about Code2City, Code2City$_{\mathrm{VR}}$, or Eclipse with respect to a given theme, we summed the scores a participant gave for the questions of that theme. For example, given the Tool Quality theme, we obtained an overall score, **TQ**, computed as the sum of the scores a participant gave for the questions from TQ1 to TQ5 (see Figure 5). It is easy to grasp that TQ assumes values in between 5 and 35—the higher the TQ value, the better the thinking of participants about the quality of the used tool (*i.e.,* Code2City, Code2City$_{\mathrm{VR}}$, or Eclipse) is. Similarly, we computed the overall scores for the other themes: **IQ** for Information Quality; **SQ** for Service Quality; **P** for Playfulness; **PEU** for Perceived Ease of Use; **PU** for Perceived Usefulness; **AU** for Attitude toward Use; and **BIU** for Behavioral Intention to Use. For any of overall score, a high value is desirable.

Rate from 1="I strongly disagree" to 7="I strongly agree" the extent to which you disagree/agree with each of the following statements (Tool is equal to: Code2City, Code2City$_{\text{VR}}$, or Eclipse; that is, it represents the tool used to fulfill the program comprehension tasks):	
Tool Quality	
TQ1. Tool has appropriate graphics.	----------
TQ2. Tool has easy navigation to information.	----------
TQ3. Tool has fast response to the user's input.	----------
TQ4. Tool has good functionality to support program comprehension tasks.	----------
TQ5. Tool is error-free.	----------
Information Quality	
IQ1. Tool has sufficient contents where I expect to find information.	----------
IQ2. Tool provides complete information.	----------
IQ3. Tool provides accurate information.	----------
IQ4. Tool provides timely information.	----------
IQ5. Tool provides reliable information.	----------
IQ6. Tool communicates information in an appropriate format.	----------
Service Quality	
SQ1. Tool instills confidence in users, reducing their uncertainty.	----------
SQ2. Tool gives a professional and competence image.	----------
Playfulness	
P1. When interacting with Tool, I did not realize the time elapsed.	----------
P2. When interacting with Tool, I was not aware of any noise.	----------
P3. I had fun when fulfilling the program comprehension tasks with Tool.	----------
P4. Using Tool to fulfill the program comprehension tasks made me happy.	----------
P5. Using Tool stimulated my curiosity to explore the program.	----------
P6. Using Tool aroused my imagination.	----------
Perceived Ease of Use	
PEU1. Learning Tool was easy for me.	----------
PEU2. It would be possible to learn to use Tool without expert help.	----------
PEU3. My interaction with Tool was clear and understandable.	----------
PEU4. It was easy for me to become skillful at using Tool.	----------
PEU5. Using Tool does not require a lot of mental effort.	----------
PEU6. I find it easy to get Tool to do what I want it to do.	----------
PEU7. I find Tool user friendly.	----------
Perceived Usefulness	
PU1. Using Tool enabled me to accomplish the program comprehension tasks quickly.	----------
PU2. Using Tool to fulfill the program comprehension tasks increased my productivity.	----------
PU3. Using Tool to fulfill the program comprehension tasks improved their correctness.	----------
PU4. Using Tool made the program comprehension tasks easy.	----------
Attitude toward Use	
AU1. Using Tool is a good idea.	----------
AU2. Using Tool is a wise idea.	----------
AU3. Using Tool is a satisfactory idea.	----------
AU4. Using Tool is a positive idea.	----------
AU5. Using Tool is an appealing idea.	----------
Behavioral Intention to Use	
BIU1. I would like to keep using Tool in the future.	----------
BIU2. I would like to use Tool on a regular basis in the future.	----------
BIU3. I would recommend others to use Tool.	----------

Fig. 5 Post-mortem survey.

5 Results and Discussion

In this section, we first present and discuss the results concerning the feelings and emotions and then the users' thinking. We also report the results from a further analysis to understand if the Code2City$_{\text{VR}}$'s users experienced some side effects due to the use of VR. Finally, we provide an overall discussion of the results and highlight the threats to the validity of our study.

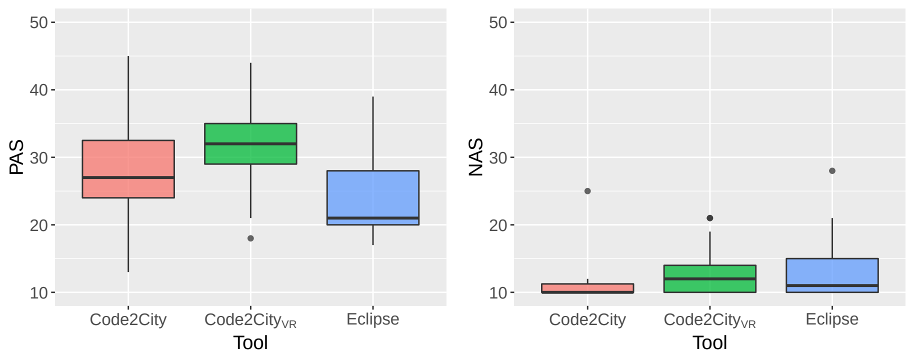

Fig. 6 Boxplots for PAS and NAS.

5.1 Feelings and Emotions

In Figure 6, we graphically summarize (by means of boxplots) the distributions of the PAS and NAS values for each tool. Some descriptive statistics for these distributions can be found in Appendix B (see Table 15). Regardless of the tool, we can notice that the distributions for PAS are higher than the ones for NAS (see Figure 6). That is, after performing the comprehension tasks with the support of Code2City, Code2City$_{\text{VR}}$, or Eclipse, the participants felt more positive affects than those negative.

In the following of this section, we first present the results about the positive affects and then those about the negative affects.

5.1.1 Positive Affects (PAS)

As shown in Figure 6, the boxplot of Code2City$_{\text{VR}}$ for PAS is much higher than the one of Eclipse so suggesting that the use of Code2City$_{\text{VR}}$ causes feelings and emotions more positive with respect to the use of Eclipse. On average, the levels of positive feelings and emotions are equal to 31.412 and 24.231 for Code2City$_{\text{VR}}$ and Eclipse, respectively (see Table 15 in Appendix B). We can also notice a similar outcome, but less marked, when comparing the boxplots of Code2City$_{\text{VR}}$ and Code2City. That is, the use of Code2City$_{\text{VR}}$ seems to lead to slightly better PAS values than the use of Code2City (mean=28.167). To understand whether the observed differences in the PAS values are significant, we planned to run the one-way ANOVA test. This parametric test is used for determining whether there are significant differences between the means of two or more independent groups (*i.e.,* Code2City, Code2City$_{\text{VR}}$, and Eclipse in our case) [41]. If there are significant differences, a post-hoc analysis is needed to understand which specific groups are significantly different (*e.g.,* Code2City$_{\text{VR}}$ vs. Eclipse). To this end, we ran the Tukey's HSD test [34]. It is worth mentioning that the ANOVA test has some assumptions that should not be violated. Therefore, before running the ANOVA test, we checked normality of data and homogeneity of variance. To check the assumption of normality, we ran the Shapiro-Wilk test [29] (Shapiro test, from here onwards) for each group (*e.g.,* the Code2City group). As for the assumption of homogeneity, we ran Bartlett's test [3]. In case of violation of the ANOVA

Table 2 Results (*i.e.,* p-values) from the ANOVA or Kruskal test, as well as those from the post-hoc analysis (HSD or Dunn's test), for PAS and NAS. [⋆]: p-value less than α.

	ANOVA/Kruskal	Post-hoc (HSD/Dunn)		
		Code2City$_{\mathrm{VR}}$-Code2City	Code2City$_{\mathrm{VR}}$-Eclipse	Code2City-Eclipse
PAS	0.04⋆ (ANOVA)	0.48	0.031⋆	0.387
NAS	0.164 (Kruskal)	–	–	–

assumptions, we planned to run its non-parametric alternative, the Kruskal-Wallis test [41] (Kruskal, from here onwards), with the corresponding post-hoc analysis test, the Dunn's test [8].[1] As it is customary with statistical hypothesis tests, we fixed the significance level, α, at 5%.

In Table 2, we report the p-value the ANOVA test returned for PAS, as well as the p-values returned by the post-hoc analysis. In particular, we can observe that the p-value for PAS is equal to 0.04, thus it is less than α=0.05. This means that there are significant differences between the PAS values of Code2City, Code2City$_{\mathrm{VR}}$, and Eclipse. This outcome justifies a post-hoc analysis by means of the HSD test. This test indicates that there are significant differences between the distributions of Code2City$_{\mathrm{VR}}$ and Eclipse (p-value=0.031), and according to the descriptive statistics (see Appendix B) or boxplots (see Figure 6), these differences are in favor of Code2City$_{\mathrm{VR}}$. In the other two pairwise comparisons (*e.g.,* Code2City vs. Code2City$_{\mathrm{VR}}$), the differences are not significant. Therefore, we can conclude that participants who had experimented Code2City$_{\mathrm{VR}}$ felt significantly better positive affects than those who had used Eclipse.

We deepened the study on the positive affects by analyzing the scores for each positive affect in the PANAS questionnaire. This is to understand which feelings/emotions caused the observed differences in the PAS values of Code2City, Code2City$_{\mathrm{VR}}$, and Eclipse. We depicted these scores by means of the (grouped) barplots in Figure 7. Each group of bars corresponds to a possible score and is made up of three bars—one for each treatment group (*e.g.,* Code2City). The height of a bar is given by the count for a given score and a given treatment group. The barplots in Figure 7 seem to indicate that the participants who used Code2City$_{\mathrm{VR}}$ gave higher scores for interested, excited, enthusiastic, inspired, and active, with respect to those who used Code2City and Eclipse.

We also performed statistical hypothesis tests to determine which differences were significant in the scores of each positive affect. To this end, we applied the same analysis procedure described previously (*i.e.,* either ANOVA or Kruskal test followed by the corresponding post-hoc analysis test). The results of this analysis are summarized in Table 3. We can notice significant differences among Code2City, Code2City$_{\mathrm{VR}}$, and Eclipse for excited (p-value=0.015), enthusiastic (p-value=0.008), and inspired (p-value=0.035). The post-hoc analysis revealed that the scores given by the Code2City$_{\mathrm{VR}}$'s users were significantly higher than those given by the Eclipse users. That is, who had used Code2City$_{\mathrm{VR}}$ felt significantly more excited, enthusiastic, and inspired than those who had experimented Eclipse.

[1] The p-values returned by the Dunn's test are adjusted according to the Bonferroni adjustment [7].

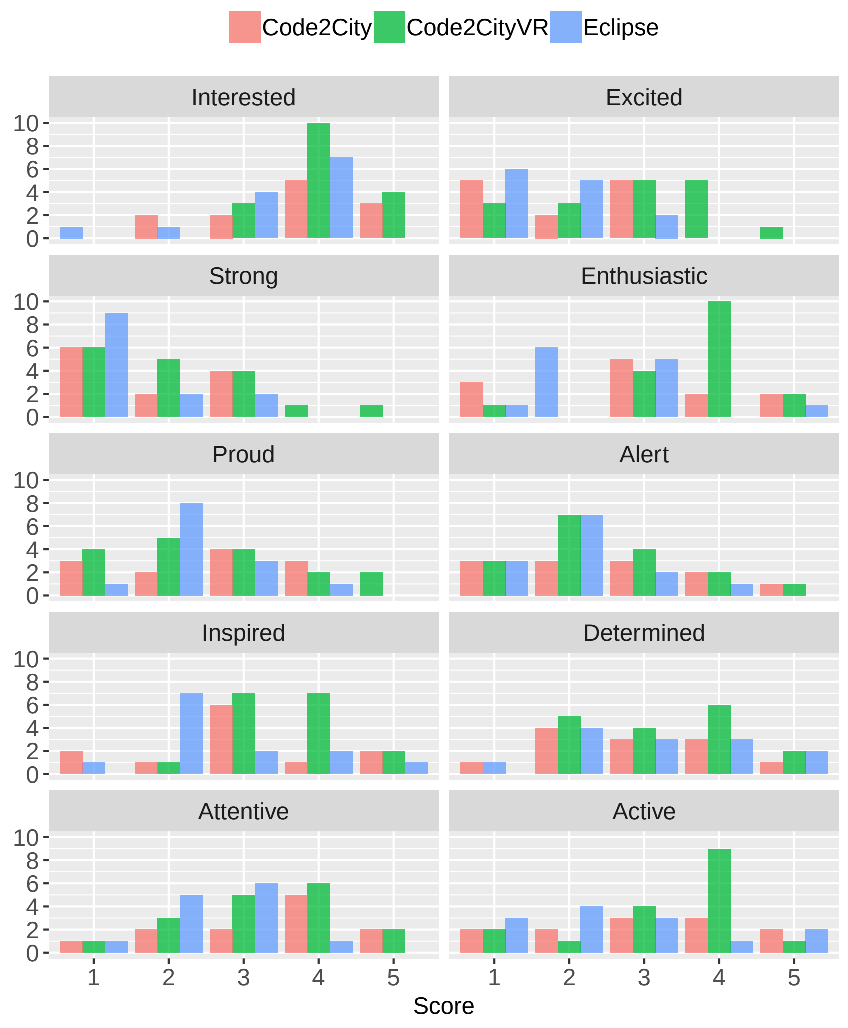

Fig. 7 Barplots for the positive affects in the PANAS questionnaire.

Table 3 Results (*i.e.*, p-values) from the ANOVA or Kruskal test, as well as those from the post-hoc analysis (HSD or Dunn's test), for the positive affects in the PANAS questionnaire [⋆]: p-value less than α.

	ANOVA/Kruskal	Post-hoc (HSD/Dunn)		
		Code2City$_{\mathrm{VR}}$-Code2City	Code2City$_{\mathrm{VR}}$-Eclipse	Code2City-Eclipse
Interested	0.098 (Kruskal)	–	–	–
Excited	0.015⋆ (Kruskal)	0.672	0.048⋆	0.31
Strong	0.183 (Kruskal)	–	–	–
Enthusiastic	0.008⋆ (Kruskal)	0.147	0.003⋆	0.306
Proud	0.799 (Kruskal)	–	–	–
Alert	0.54 (Kruskal)	–	–	–
Inspired	0.035⋆ (Kruskal)	0.225	0.015⋆	0.472
Determined	0.682 (Kruskal)	–	–	–
Attentive	0.0831 (ANOVA)	–	–	–
Active	0.268 (Kruskal)	–	–	–

5.1.2 Negative Affects (NAS)

As for NAS, we can notice no huge difference among the boxplots of Code2City, Code2City$_{\mathrm{VR}}$, and Eclipse (see Figure 6). On average, the levels of negative feelings and emotions of the those who had used Code2City, Code2City$_{\mathrm{VR}}$, and Eclipse are equal to 11.667, 13.235, and 13.538, respectively (see Appendix B).

To have a confirmation that there is no significant difference in the NAS values of Code2City, Code2City$_{\mathrm{VR}}$, and Eclipse, we did not apply the ANOVA test because its assumptions were violated. Therefore, we ran the Kruskal test. As shown in Table 2, the p-value (0.164) the Kruskal test returned does not indicate significant differences in the NAS values of Code2City, Code2City$_{\mathrm{VR}}$, and Eclipse. This outcome does not justify a post-hoc analysis.

Similar to the analysis done for the positive affects, we analyzed the scores for each negative affect by means of (grouped) barplots and statistical hypothesis tests. As shown by the barplots in Figure 8, we can observe that there is no noticeable difference among Code2City, Code2City$_{\mathrm{VR}}$, and Eclipse for any negative affect in the PANAS questionnaire. This is confirmed by the results from the statistical hypothesis tests we ran, which are summarized in Table 4. That is, none of these tests returned a p-value less than α so suggesting that using Code2City, Code2City$_{\mathrm{VR}}$, or Eclipse does not influence any negative affect.

5.2 Thinking

In Figure 9, we provide the boxplots depicting the distributions for TQ, IQ, SQ, P, PEU, PU, AU, and BIU grouped by tool. The descriptive statistics for these distributions are available in Appendix B (see Table 16).

We applied the same data analysis procedure as that used in the previous subsection. That is, we first leveraged exploratory data analysis (*i.e.,* boxplots or barplots) with the support of some descriptive statistics. Then we applied the ANOVA or Kruskal test. When needed (*i.e.,* in case of significant differences), we ran the corresponding post-hoc test (*i.e.,* HSD or Dunn's test). The results from the statistical hypothesis tests are summarized in Table 5.

In the following of this subsection, we present the participants' thinking about Code2City, Code2City$_{\mathrm{VR}}$, and Eclipse according to eight themes of the post-mortem survey.

5.2.1 Tool Quality (TQ)

As shown in Figure 9, there is not a huge difference in the TQ values for the considered tools. The boxplots are quite high (the highest possible value for TQ is 35) and they overlap one another. However, the TQ values of Code2City seem to be slightly better than the ones of Eclipse. The values of the descriptive statistics seem to confirm that there is not a huge difference among Code2City, Code2City$_{\mathrm{VR}}$, and Eclipse with respect to TQ; for example, the mean TQ values are 27, 26.471, and 25.385, respectively (see Appendix B). A further confirmation is provided by the results of the Kruskal test (the assumptions to apply the ANOVA test were not met). The p-values (0.158 as shown in Table 5) is greater than α, thus it does not suggest significant differences in the TQ values among Code2City,

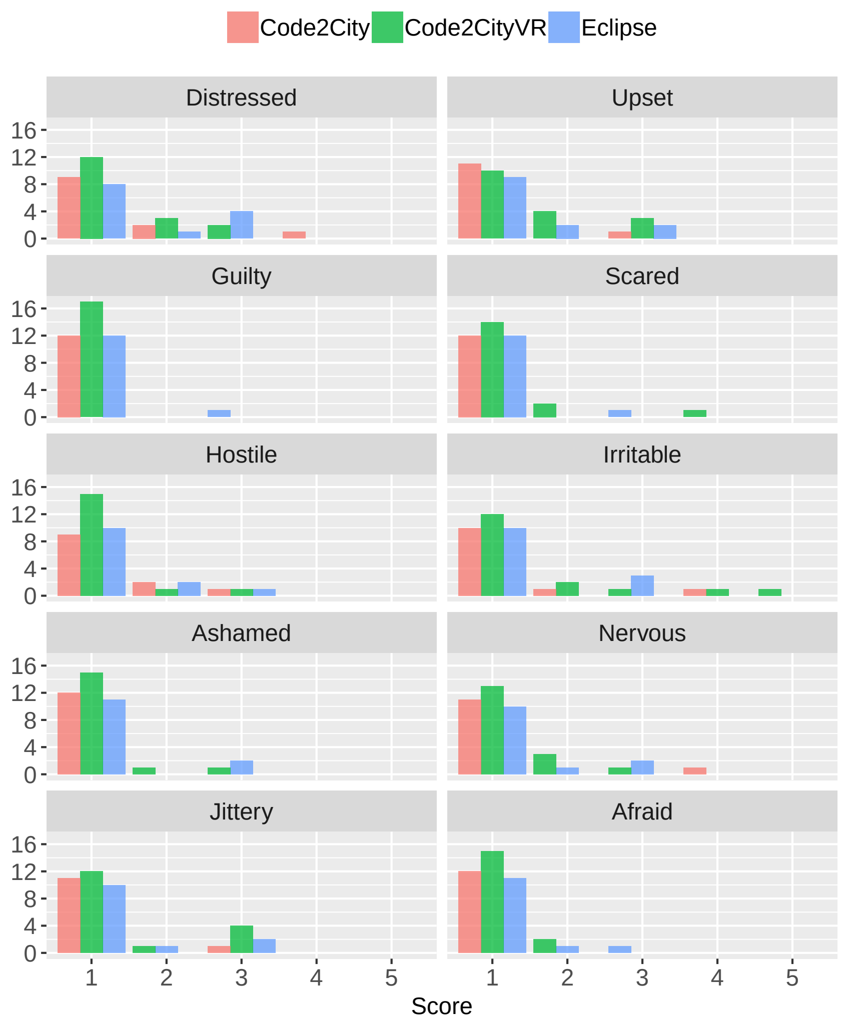

Fig. 8 Barplots for the negative affects in the PANAS questionnaire.

Table 4 Results (*i.e.,* p-values) from the ANOVA or Kruskal, as well as those from the post-hoc analysis (HSD or Dunn's test), for the negative affects in the PANAS questionnaire. [⋆]: p-value less than α.

	ANOVA/Kruskal	Post-hoc (HSD/Dunn)		
		Code2City$_{\text{VR}}$-Code2City	Code2City$_{\text{VR}}$-Eclipse	Code2City-Eclipse
Distressed	0.656 (Kruskal)	–	–	–
Upset	0.207 (Kruskal)	–	–	–
Guilty	0.328 (Kruskal)	–	–	–
Scared	0.287 (Kruskal)	–	–	–
Hostile	0.64 (Kruskal)	–	–	–
Irritable	0.726 (Kruskal)	–	–	–
Ashamed	0.392 (Kruskal)	–	–	–
Nervous	0.631 (Kruskal)	–	–	–
Jittery	0.414 (Kruskal)	–	–	–
Afraid	0.392 (Kruskal)	–	–	–

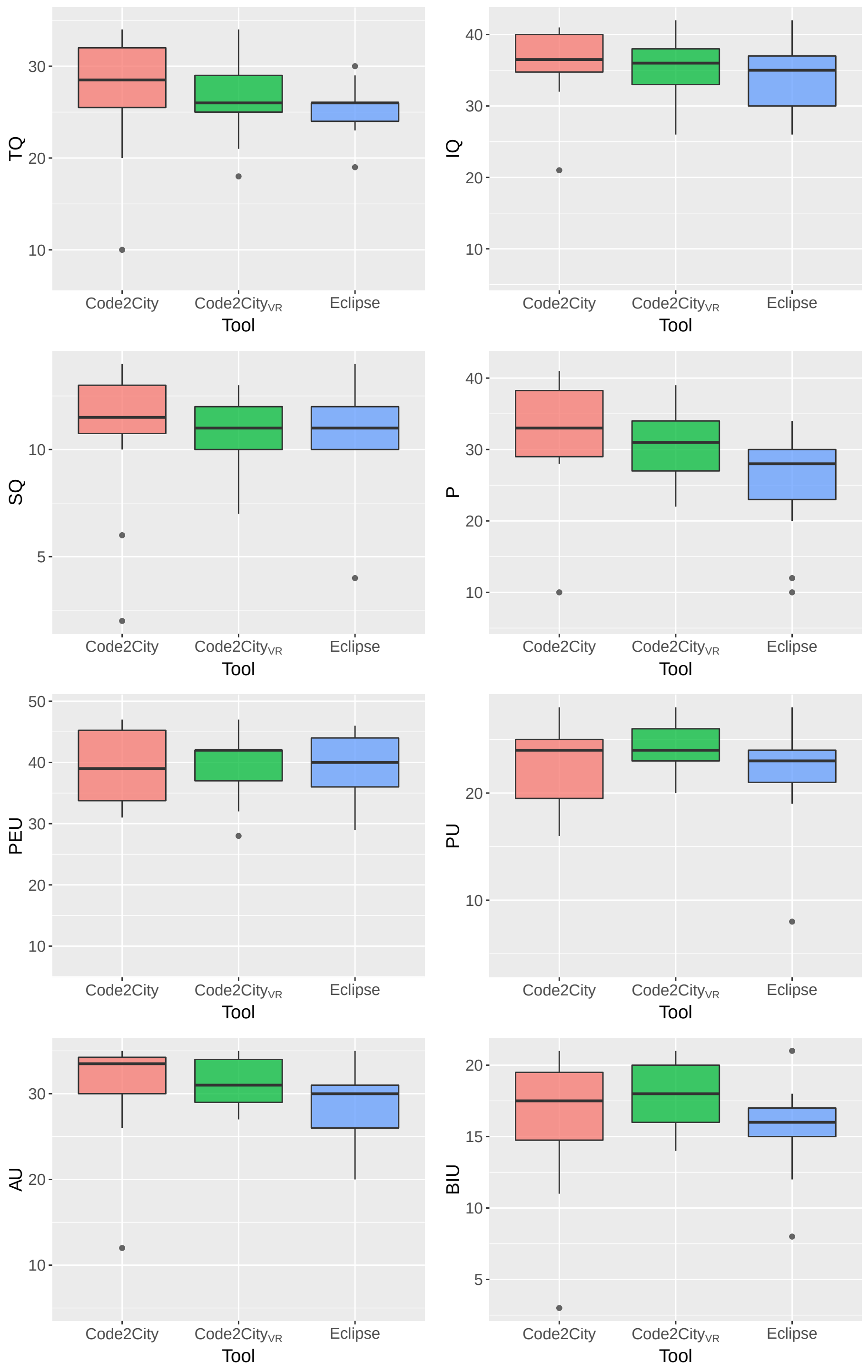

Fig. 9 Boxplots for TQ, IQ, SQ, P, PEU, PU, AU, and BIU, grouped by tool.

Table 5 Results (*i.e.,* p-values) from the ANOVA or Kruskal test, as well as those from the post-hoc analysis (HSD or Dunn's test), for TQ, IQ, SQ, P, PEU, PU, AU, and BIU. [*]: p-value less than α.

	ANOVA/Kruskal	Post-hoc (HSD/Dunn)		
		Code2City$_{\text{VR}}$-Code2City	Code2City$_{\text{VR}}$-Eclipse	Code2City-Eclipse
TQ	0.158 (Kruskal)	–	–	–
IQ	0.54 (Kruskal)	–	–	–
SQ	0.823 (Kruskal)	–	–	–
P	0.024* (Kruskal)	0.448	0.097	0.011*
PEU	0.912 (ANOVA)	–	–	–
PU	0.19 (Kruskal)	–	–	–
AU	0.254 (Kruskal)	–	–	–
BIU	0.344 (Kruskal)	–	–	–

Code2City$_{\text{VR}}$, and Eclipse. That is, the participants who had used Code2City$_{\text{VR}}$ judged positively the quality of this tool and it was comparable to the quality of Code2City and Eclipse.

This outcome does not exclude the presence of differences among the participants' answers to the single questions of the Tool Quality theme. We summarize these answers by means of the (grouped) barplots in Figure 10, where we can note that some participants who had experimented Code2City$_{\text{VR}}$ gave negative scores (*i.e.,* 2 or 3) to TQ1—*Tool has appropriate graphics*—while the participants who had used Code2City gave more positive scores. We executed the testing of statistical hypothesis to determine if these observed differences were significant. The

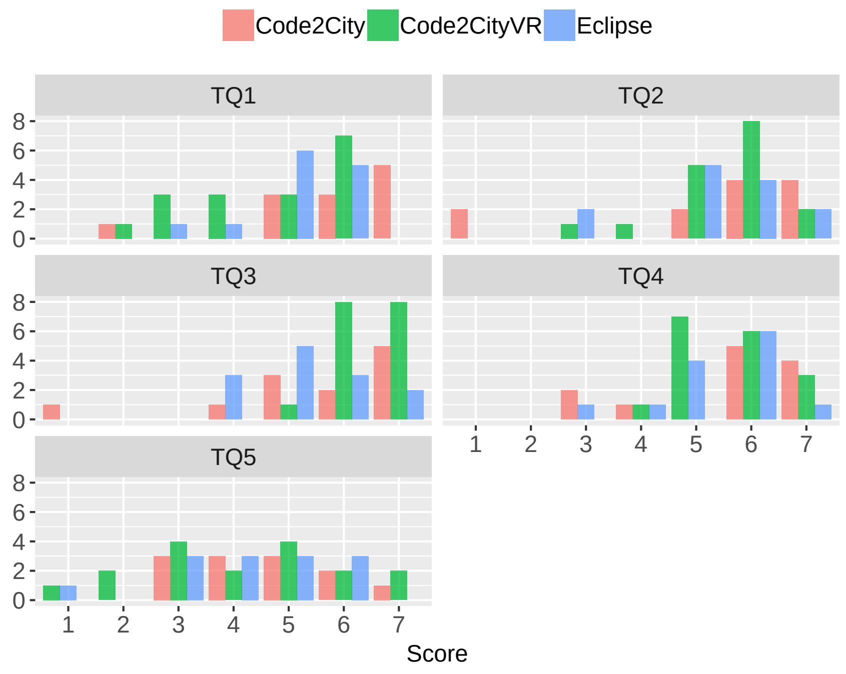

Fig. 10 Barplots for the questions about the Tool Quality theme in the post-mortem survey.

Table 6 Results (*i.e.,* p-values) from the ANOVA or Kruskal (*i.e.,* Kruskal) test, as well as those from the post-hoc test (HSD or Dunn's test), for the questions about the Tool Quality theme. [⋆]: p-value less than α.

	ANOVA/Kruskal	Post-hoc (HSD/Dunn)		
		Code2City$_{VR}$-Code2City	Code2City$_{VR}$-Eclipse	Code2City-Eclipse
TQ1	0.048⋆ (Kruskal)	0.023⋆	0.847	0.121
TQ2	0.628 (Kruskal)	–	–	–
TQ3	0.02⋆ (Kruskal)	0.264	0.008⋆	0.294
TQ4	0.532 (Kruskal)	–	–	–
TQ5	0.772 (ANOVA)	–	–	–

results from these tests are reported in Table 6. The p-value of the Kruskal test is 0.048, while the p-value of the Dunn's test comparing Code2City and Code2City$_{VR}$ is 0.023. These results suggest that the Code2City's users significantly rated more positively the graphics of the used tool with respect to the Code2City$_{VR}$ users.

As for TQ2—*Tool has easy navigation to information*—, we can observe that most of the participants gave high scores whatever the tool was (see Figure 10). No statistical difference was suggested by the Kruskal test (see Table 6).

By looking at the barplot for TQ3—*Tool has fast response to the user's input*—, we can observe scores much more positive for Code2City$_{VR}$ than for the other tools. Thanks to the p-value (0.02) returned by the Kruskal test as well as the p-values (*e.g.,* 0.008 when comparing Code2City$_{VR}$ with Eclipse) returned by the post-hoc test, we can strengthen this finding. That is, the response of Code2City$_{VR}$ was perceived significantly faster than the response of Eclipse.

As for TQ4—*Tool has good functionality to support program comprehension tasks*—and TQ5—*Tool is error-free*— we can notice no huge difference among the tools. The results in Table 6 go towards this direction because the p-values for TQ4 and TQ5 are both greater than α (*i.e.,* they do not suggest significant difference). Therefore, any of the considered tools seems to well-support program comprehension tasks since most of the scores are positive. However, the results for TQ5 seem to suggest that the participants misunderstood this statement or experienced some bugs while using either Code2City Code2City$_{VR}$ or Eclipse. The latter justification seems implausible because Code2City and Code2City$_{VR}$ were largely used and tested especially on FindBugs and if failures had occurred we would have fixed them. As for Eclipse, it is well known for its reliability. Then, the most plausible justification seems to be that a few students (equally distributed among the three groups) misunderstood TQ5. For example, we can speculate that participants could have confused the presence of bad smell (see Table 14 in Appendix A) in the FindBugs source code with errors. Taking into account the considerations before, we cannot fully trust the outcome from TQ5.

5.2.2 Information Quality (IQ)

The participants' thinking about the information quality of Code2City$_{VR}$ seems to be positive and similar to the information quality of Code2City or Eclipse—the boxplots for IQ (see Figure 9) are high and not so different one another as well as the descriptive statistics values (*e.g.,* the mean IQ values of Code2City, Code2City$_{VR}$, and Eclipse are 35.833, 35.294, and 34.154, respectively, as reported in Appendix B). We ran the Kruskal test to confirm this finding. The p-value

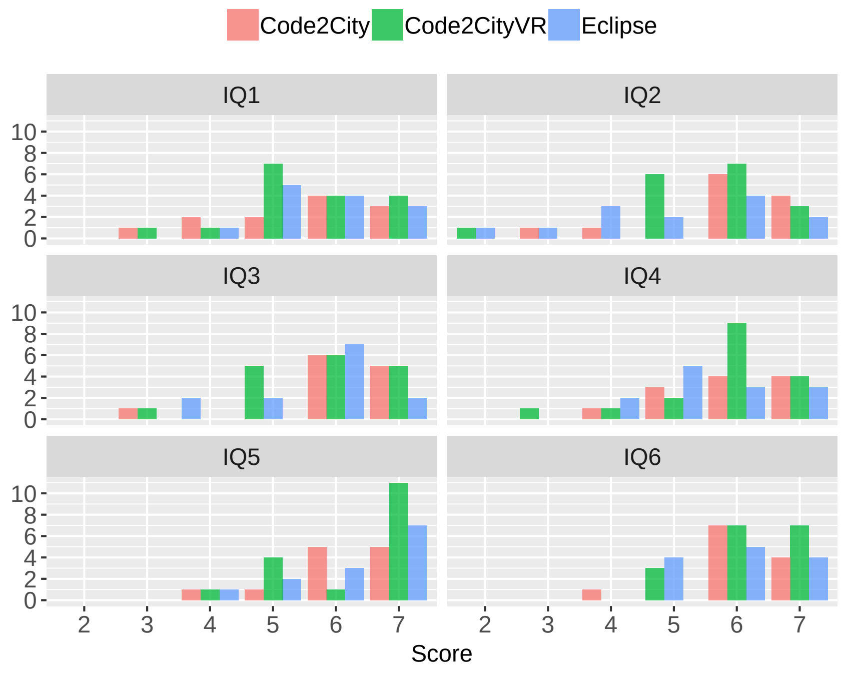

Fig. 11 Barplots for the questions about the Information Quality theme in the post-mortem survey.

Table 7 Results (*i.e.,* p-values) from the ANOVA or Kruskal test, as well as those from the post-hoc test (HSD or Dunn's test), for the questions about the Information Quality theme. [⋆]: p-value less than α.

	ANOVA/Kruskal	Post-hoc (HSD/Dunn)		
		$\text{Code2City}_{\text{VR}}$-Code2City	$\text{Code2City}_{\text{VR}}$-Eclipse	Code2City-Eclipse
IQ1	0.95 (Kruskal)	–	–	–
IQ2	0.184 (Kruskal)	–	–	–
IQ3	0.275 (Kruskal)	–	–	–
IQ4	0.546 (Kruskal)	–	–	–
IQ5	0.801 (Kruskal)	–	–	–
IQ6	0.683 (Kruskal)	–	–	–

(0.54) reported in Table 5 does not indicate significant differences with respect to the information quality of Code2City, $\text{Code2City}_{\text{VR}}$, and Eclipse. Therefore, we can conclude that the participants' thinking about the information quality of the used tool was comparable.

The barplots in Figure 11 suggest that the greater part of the participants believed that the used tool (Code2City, $\text{Code2City}_{\text{VR}}$, or Eclipse) had sufficient contents where they expected to find information (IQ1). Moreover, that tool provided complete, accurate, timely, and reliable information (IQ2, IQ3, IQ4, and IQ5). Finally, it communicated information in an appropriate format (IQ6). We can confirm these findings by means of the results from the statistical hypothesis tests (see Table 7) because all the p-values are greater than α.

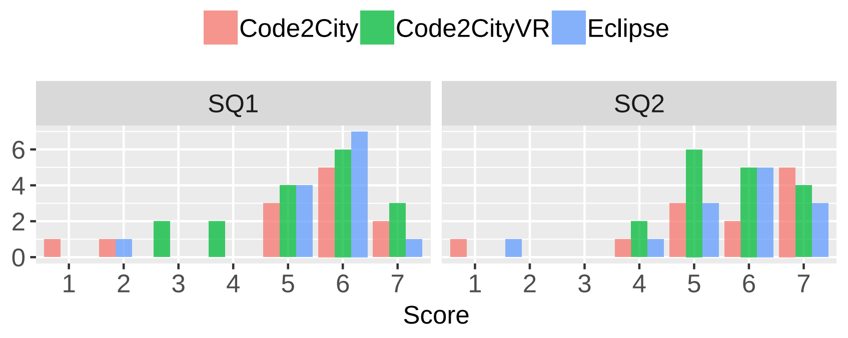

Fig. 12 Barplots for the questions about the Service Quality theme in the post-mortem survey.

Table 8 Results (*i.e.,* p-values) from the ANOVA or Kruskal (*i.e.,* Kruskal) test, as well as those from the post-hoc test (HSD or Dunn's test), for the questions about the Service Quality theme. [*]: p-value less than α.

	ANOVA/Kruskal	Post-hoc (HSD/Dunn's test)		
		Code2City$_{\mathrm{VR}}$-Code2City	Code2City$_{\mathrm{VR}}$-Eclipse	Code2City-Eclipse
SQ1	0.972 (Kruskal)	–	–	–
SQ2	0.887 (Kruskal)	–	–	–

5.2.3 Service Quality (SQ)

The boxplots in Figure 9 indicate that the distribution of SQ for Code2City$_{\mathrm{VR}}$ is quite similar to the other distributions. On average, these values are also high (see Appendix B): 10.75, 11, and 11 for Code2City, Code2City$_{\mathrm{VR}}$, and Eclipse, respectively—the best possible value is 14. The p-value returned by the Kruskal test (0.823, see Table 5) is greater than α. That is, the service quality of Code2City$_{\mathrm{VR}}$ was judged positively and is it was similar to the service quality of Code2City or Eclipse.

In Figure 12, we summarize the answers to the questions about the service quality of the tools. We can observe that, regardless of the tool, most of the participants pointed out that it instilled confidence and reduce their uncertainty when they performed the program comprehension tasks (SQ1). A similar result can be observed for SQ2—*Tool gives a professional and competence image*. That is, most of the scores for SQ2 were positive.

By considering the p-values in Table 8, we can confirm that the participants similarly rated the service quality of the tools they used to perform the program comprehension tasks. This is because the p-values are greater than α.

5.2.4 Playfulness (P)

As shown in Figure 9, there could be differences in the P values among Code2City, Code2City$_{\mathrm{VR}}$, and Eclipse. These differences are more marked when comparing Code2City with Eclipse—the boxplot for Code2City is higher than the one for Eclipse. On average, the values of P are 32.417 and 25.308, respectively (see Appendix B). Also between Code2City$_{\mathrm{VR}}$ and Eclipse there could be differences in

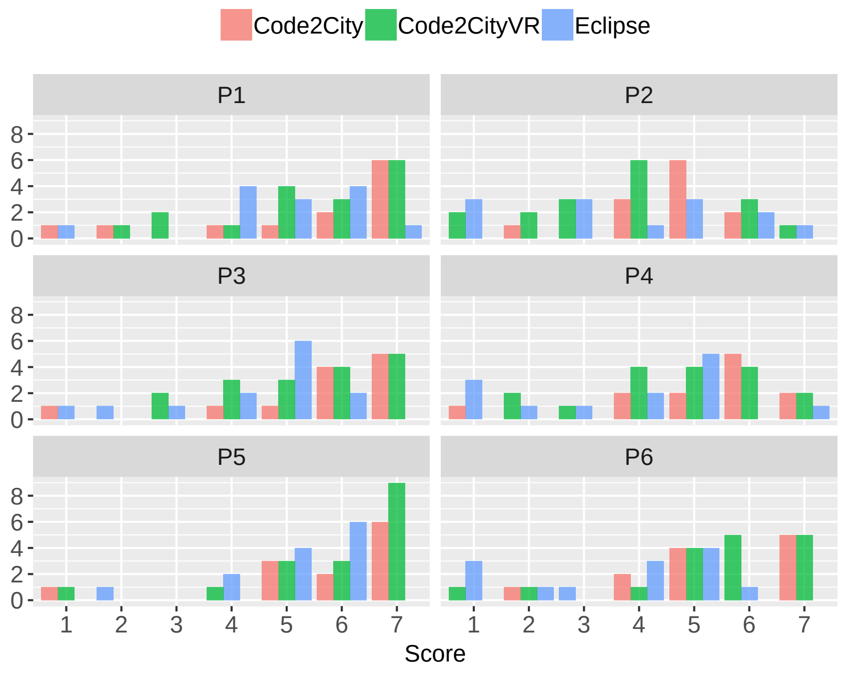

Fig. 13 Barplots for the questions about Playfulness in the post-mortem survey.

Table 9 Results (*i.e.,* p-values) from the ANOVA or Kruskal test, as well as those from the post-hoc analysis (HSD or Dunn's test), for the questions about Playfulness. [⋆]: p-value less than α.

	ANOVA/Kruskal	Post-hoc (HSD/Dunn)		
		Code2City$_{\mathrm{VR}}$-Code2City	Code2City$_{\mathrm{VR}}$-Eclipse	Code2City-Eclipse
P1	0.275 (Kruskal)	–	–	–
P2	0.278 (Kruskal)	–	–	–
P3	0.026⋆ (Kruskal)	0.51	0.088	0.013⋆
P4	0.069 (Kruskal)	–	–	–
P5	0.043⋆ (Kruskal)	1	0.029⋆	0.067
P6	0.008⋆ (Kruskal)	1	0.006⋆	0.019⋆

the P values, but such differences are less marked. For example, the mean value of P for Code2City$_{\mathrm{VR}}$ is 30.706. We can also note that the values of P for Code2City and Code2City$_{\mathrm{VR}}$ are quite high (the best possible value is 42).

The Kruskal and Dunn's tests, whose results are reported in Table 5, suggest that there are significant differences between the P distributions for Code2City, Code2City$_{\mathrm{VR}}$, and Eclipse (p-value=0.024). Moreover, the distributions of Code2City and Eclipse significantly differ (p-value=0.011), while those of Code2City$_{\mathrm{VR}}$ and Eclipse does not (p-value=0.097). Summing up, the playfulness of Eclipse is significantly worse than the playfulness of Code2City and it is slightly worse than the one of Code2City$_{\mathrm{VR}}$.

In Figure 13, we summarize the answers to the questions of the Playfulness theme. Most of the participants who used Code2City or Code2City$_{\mathrm{VR}}$ admitted not to realize the passing of time (P1)—most of the scores are 6 and 7. The participants who experimented Eclipse seems to agree slightly less with the statement of the question P1. By looking at the results of the statistical hypothesis tests in Table 9 (*i.e.,* those about the Playfulness theme), we cannot say that the observed differences in the answers to P1 are significant (p-value=0.275).

As for P2—*When interacting with Tool, I was not aware of any noise*—, the participants seem to slightly agree with P2 regardless of the tool. The p-value (0.278) returned by the Kruskal seems to confirm that there is no difference due to the used tool.

The Code2City and Code2City$_{\mathrm{VR}}$'s users seemed to have more fun than the Eclipse's users (P3)—most of the participants, who used Code2City or Code2City$_{\mathrm{VR}}$, gave very positive scores (see Figure 13). The p-value (0.026) returned by the Kruskal test indicates a significant difference in the answers to P3, but this difference is not significant when contrasting Code2City$_{\mathrm{VR}}$ with Eclipse (p-value=0.088), while it is significant when contrasting Code2City with Eclipse (p-value=0.013).

By considering the answers to P4—*Using Tool to fulfill the program comprehension tasks made me happy*—, it is easy to notice that only one Eclipse user gave very positive scores (*i.e.,* 6 or 7) for this question, while the use of Code2City and Code2City$_{\mathrm{VR}}$ to perform the tasks seemed to make the participants happier. However, these observed differences in the answers to P4 appear not to be significant (p-value 0.069).

The barplot for P5 suggests that Code2City and Code2City$_{\mathrm{VR}}$ stimulated more the curiosity of their users to explore the program than Eclipse (see Figure 13). We can strengthen this finding thanks to the results of the Kruskal test (p-values=0.043). Moreover, the Code2City$_{\mathrm{VR}}$'s users were significantly more stimulated to explore the program than those using Eclipse (p-value=0.029)—no significant difference was observed between the answers of the Code2City and Eclipse's users (p-value=0.067).

The barplot for P6 indicates that the use of Code2City and Code2City$_{\mathrm{VR}}$ aroused more the imagination of their users with respect to Eclipse. The tests we run confirm with strength this finding. In fact, the p-value (0.008) of the Kruskal test was less than α as well as the p-values returned by the Dunn's test for Code2City vs. Eclipse (p-value=0.019) and Code2City$_{\mathrm{VR}}$ vs. Eclipse (p-value=0.019). That is, the use of Code2City and Code2City$_{\mathrm{VR}}$ aroused the imagination of their users significantly more than the use of Eclipse.

5.2.5 Perceived Ease of Use (PEU)

We can observe in Figure 9 that the distributions of the PEU values are quite similar to one another. This suggests that the perceived ease of use of Code2City$_{\mathrm{VR}}$ is similar to that of Code2City or Eclipse. This finding is confirmed by both the descriptive statistic values—*e.g.,* on average, the PEU value is 39.417 for Code2City, 39.529 for Code2City$_{\mathrm{VR}}$, and 38.692 for Eclipse (see Appendix B)—and the p-value (0.912) returned by the ANOVA test (see Table 5).

The answers to the questions about the perceived ease of use of the tools are depicted in Figure 14, while we summarize in Table 10 the results of the statistical hypothesis tests we ran for these questions.

The answers to PEU1 suggest that the participants found it easy to learn Code2City, Code2City$_{\mathrm{VR}}$, or Eclipse (see Figure 14). Moreover, it seems that

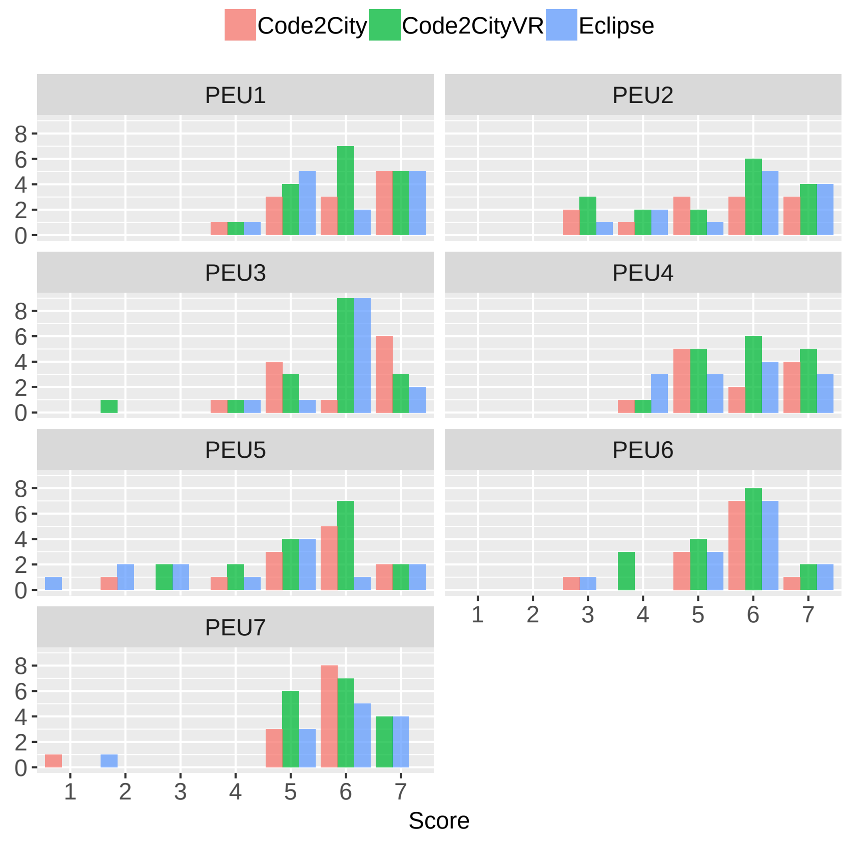

Fig. 14 Barplots for for the questions about Perceived Ease of Use in the post-mortem survey.

Table 10 Results (*i.e.*, p-values) from the ANOVA or Kruskal test, as well as those from the post-hoc analysis (HSD or Dunn's test), for the questions about Perceived Ease of Use. [*]: p-value less than α.

	ANOVA/Kruskal	Post-hoc (HSD/Dunn)		
		Code2City$_{\mathrm{VR}}$-Code2City	Code2City$_{\mathrm{VR}}$-Eclipse	Code2City-Eclipse
PEU1	0.919 (Kruskal)	–	–	–
PEU2	0.747 (Kruskal)	–	–	–
PEU3	0.718 (Kruskal)	–	–	–
PEU4	0.7 (Kruskal)	–	–	–
PEU5	0.156 (Kruskal)	–	–	–
PEU6	0.82 (Kruskal)	–	–	–
PEU7	0.543 (Kruskal)	–	–	–

they believed that it would be possible to learn to use such tools without the help of an expert (PEU2). Regardless of the tool, the participants also agreed with the statements PEU3—*My interaction with Tool was clear and understandable*—and PEU4—*It was easy for me to become skillful at using Tool.* As for PEU5, it seems that using Code2City, Code2City$_{\mathrm{VR}}$, or Eclipse does not require a lot of mental effort. The answers for the statements PEU6—*I find it easy to get Tool to do what*

I want it to do—and PEU7—*I find Tool user friendly*—-were mostly positive for any considered tool.

The results in Table 10 seem to confirm that there is no difference among the answers of the participants that experimented Code2City, Code2City, or Eclipse. That is, the statistical hypothesis tests revealed no significant difference for the questions from PEU1 to PEU7.

5.2.6 Perceived Usefulness (PU)

The boxplots in Figure 9 suggest that the perceived usefulness of Code2City, Code2City$_{\mathrm{VR}}$, and Eclipse was almost similar although that of Code2City$_{\mathrm{VR}}$ was slightly better—the boxplot for Code2City$_{\mathrm{VR}}$ is shorter and higher. On average, the perceived usefulness quantified by means of PU is 22.333, 24.471, and 21.923, respectively (see Appendix B). The results in Table 5 seem to confirm that there is no significant difference when comparing the perceived usefulness of Code2City, Code2City$_{\mathrm{VR}}$, and Eclipse (p-value=0.19).

As summarized by the barplots in Figure 15, we can notice small differences in the answers to PU1—*Using Tool enabled me to accomplish the program comprehension tasks quickly*. In particular, the Code2City$_{\mathrm{VR}}$'s users believed to accomplish the tasks slightly faster than the Code2City' and Eclipse users. By looking at the results in Table 11, we can confirm this finding. This is, the observed differences are not significant since the p-value is 0.442.

As for PU2—*Using Tool to fulfill the program comprehension tasks increased my productivity*—, whatever the tool was, the participants believed that the tool they experimented increased their productivity. The p-value (0.771) returned by the Kruskal test confirms that there is no significant difference.

By observing the answers to PU3—*Using Tool to fulfill the program comprehension tasks improved their correctness*—we can notice that the participant using

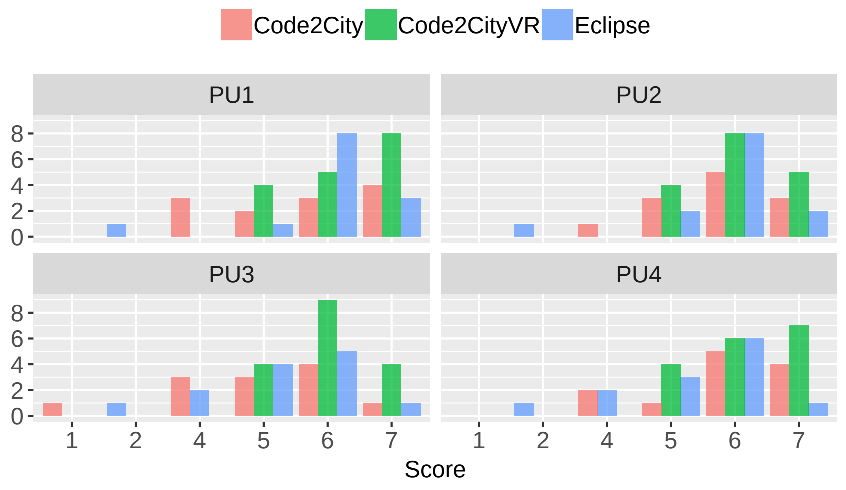

Fig. 15 Barplots for the questions about the Perceived Usefulness theme in the post-mortem survey.

Table 11 Results (*i.e.,* p-values) from the ANOVA or Kruskal test, as well as those from the post-hoc analysis (HSD or Dunn's test), for the questions about the Perceived Usefulness theme. [⋆]: p-value less than α.

	ANOVA/Kruskal	Post-hoc (HSD/Dunn)		
		Code2City$_{\text{VR}}$-Code2City	Code2City$_{\text{VR}}$-Eclipse	Code2City-Eclipse
PU1	0.442 (Kruskal)	–	–	–
PU2	0.771 (Kruskal)	–	–	–
PU3	0.043⋆ (Kruskal)	0.037⋆	0.076	1
PU4	0.089 (Kruskal)	–	–	–

Code2City$_{\text{VR}}$ were mostly convinced to improve the correctness of the program comprehension tasks because of Code2City$_{\text{VR}}$. This pattern is not so strong for Code2City nor Eclipse. Thanks to the results of the statistical hypothesis tests, we can say that there are significant differences in the answers to PU3 (p-value=0.043) and that the Code2City$_{\text{VR}}$'s users were convinced to improve the correctness of the tasks significantly more than the Code2City's users (p-value=0.037).

Finally, regardless of the tool, the participants thought that thanks to the used tool the program comprehension tasks were easy (PU4—*Using Tool made the program comprehension tasks easy*). This finding seems not to depend on the used tool since the p-value of the Kruskal test is 0.089.

5.2.7 Attitude towards the Use (AU)

The participants' thinking about the attitude towards the use of Code2City$_{\text{VR}}$ was comparable to that of Code2City or Eclipse because the boxplots in Figure 9 overlap (although that of Eclipse is slightly lower). The ran tests suggest that there is no significant difference in the thinking of the participants about this theme (p-value=0.254 as shown in Table 5).

In Figure 16, we can observe the barplots depicting the answers to AU1 to AU5. Thanks to these barplots we can notice that the most of the Code2City$_{\text{VR}}$ user considered the use of this tool a good wise, satisfactory, and positive idea (AU1, AU2, AU3, and AU4, respectively). A similar observation can be done for Code2City or Eclipse. That is, whatever the tool was, the participants mostly agreed with the statements from AU1 to AU4 were positive. The results of the statistical hypothesis test we report in Table 12 confirm that there is no difference in the participants' answers from AU1 to AU4 (p-values less than α).

As for the AU5, the barplot suggests that the Eclipse's users do not consider the use of Eclipse as an appealing idea as much as the Code2City of Code2City$_{\text{VR}}$ users. The p-values of the Kruskal test (0.016), as well as the p-values of the Dunn's tests (0.046 when contrasting Code2City$_{\text{VR}}$ and Eclipse, and 0.009 when contrasting Code2City and Eclipse), allow strengthening this finding.

5.2.8 Behavioral Intention to Use (BU)

The boxplots for BIU (see Figure 9) overlap one another although the boxplot of Eclipse is slightly lower than the one of Code2City$_{\text{VR}}$. On average, the BIU values are 17.588 for Code2City$_{\text{VR}}$ while 16.167 and 15.692 for Code2City and Eclipse

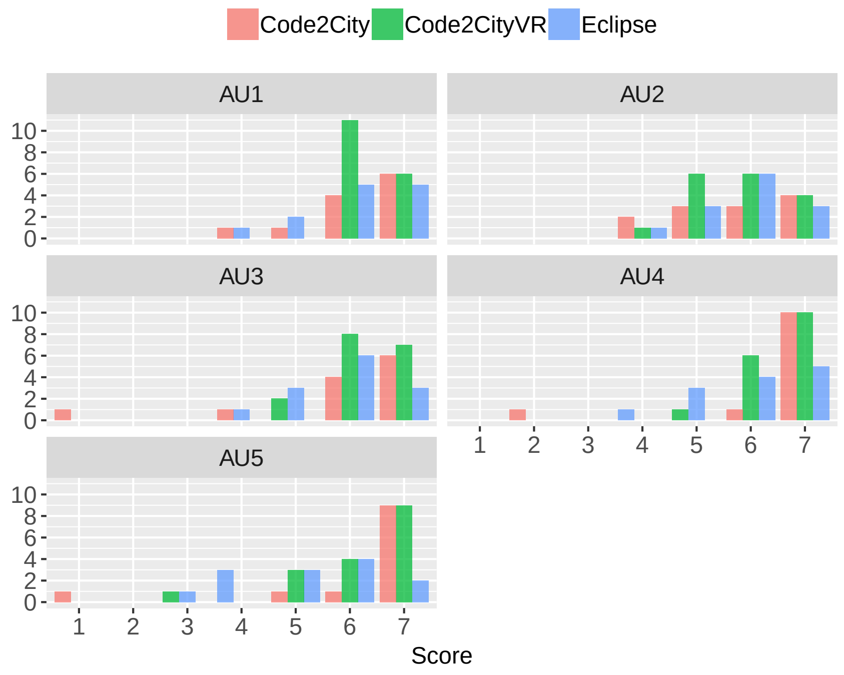

Fig. 16 Barplots for the questions about the Attitude Toward the Use theme in the post-mortem survey.

Table 12 Results (*i.e.,* p-values) from the ANOVA or Kruskal test, as well as those from the post-hoc analysis (HSD or Dunn's test), for the questions about the Attitude Toward the Use theme. [⋆]: p-value less than α.

	ANOVA/Kruskal	Post-hoc (HSD/Dunn)		
		Code2City$_{VR}$-Code2City	Code2City$_{VR}$-Eclipse	Code2City-Eclipse
AU1	0.789 (Kruskal)	–	–	–
AU2	0.961 (Kruskal)	–	–	–
AU3	0.335 (Kruskal)	–	–	–
AU4	0.077 (Kruskal)	–	–	–
AU5	0.016⋆ (Kruskal)	0.641	0.046⋆	0.009⋆

(see Appendix B). The p-value (0.344) reported in Table 5 did not indicate a significant difference in the BUI values.

The barplots depicting the answers to the questions from BIU1 to BIU3 are reported in Figure 17, while the results of the statistical hypothesis tests are summarized in Table 13.

Most of Code2City$_{VR}$'s users would like to keep using this tool in the future (BIU1). They would also like to use Code2City$_{VR}$ on a regular basis in the future (BIU2). By comparing the answers to BIU1 and BIU2 the Code2City$_{VR}$'s users gave with respect to those of the other users, we can obverse no noticeable difference. The p-values (0.309 for BIU1 and 0.471 for BIU2) of the statistical hypothesis tests confirm these findings. Finally, the greater part of the partici-

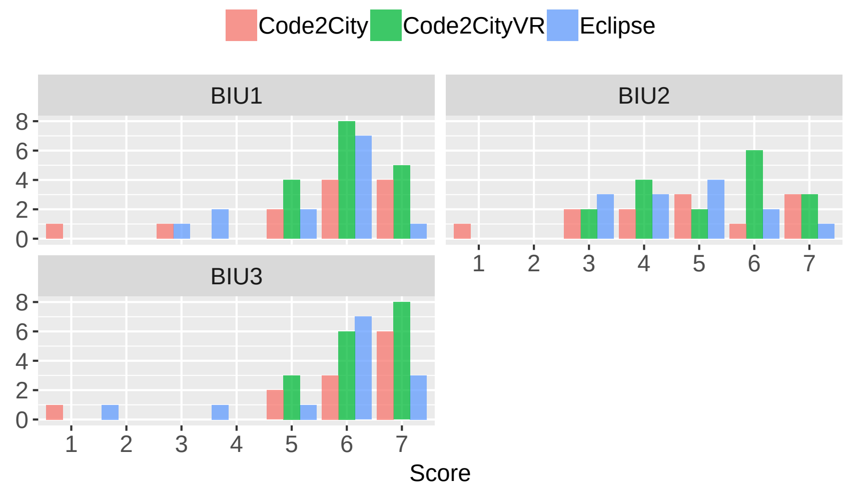

Fig. 17 Barplots for the questions about about the Behavioral Intention to Use theme in the post-mortem survey.

Table 13 Results (*i.e.*, p-values) from the ANOVA or Kruskal test, as well as those from the post-hoc analysis (HSD or Dunn's test), for the questions about the Behavioral Intention to Use theme. [$\star$]: p-value less than α.

	ANOVA/Kruskal	Post-hoc (HSD/Dunn)		
		Code2City$_{\mathrm{VR}}$-Code2City	Code2City$_{\mathrm{VR}}$-Eclipse	Code2City-Eclipse
BIU1	0.309 (Kruskal)	–	–	–
BIU2	0.471 (Kruskal)	–	–	–
BIU3	0.442 (Kruskal)	–	–	–

pants would recommend the used their tool (BIU3). Again, there is no significant difference in the answers to BIU3 (p-value=0.442).

5.3 Side Effects of Virtual Reality: Results from a Further Analysis

VR is known to cause physical discomforts when people deal with it for some time. To understand if the Code2City$_{\mathrm{VR}}$'s users experienced or not such side effects, we asked them to report the experienced physical discomforts just after using Code2City$_{\mathrm{VR}}$. We graphically summarize the experienced side effects in Figure 18 through a barplot. A few participants observed more than one side effect. Only 3 participants (out of 17) did not experience any side effect. Most of them stated to have headache, nausea, or visual annoyance after using Code2City$_{\mathrm{VR}}$. Therefore, a word of warning is needed when using VR although it positively affects users' feelings, emotions, and thinking. The outcomes of our study seem to suggest and justify future research on this point.

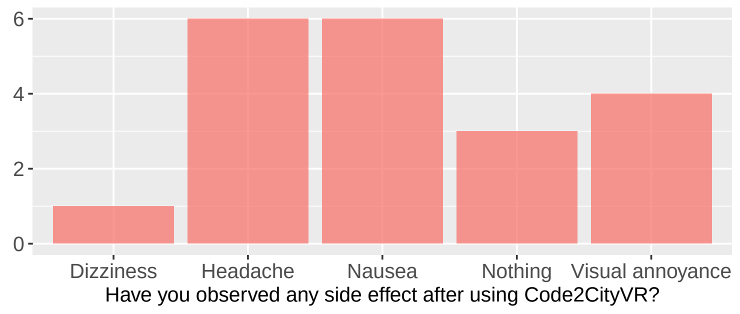

Fig. 18 Barplot on the side effects due to the use of Code2City$_{\rm VR}$.

5.4 Overall Discussion

On the basis of the results, we conclude that the use of Code2City$_{\rm VR}$ induces significantly better positive feelings and emotions as compared with Eclipse. Users are more excited, enthusiastic, and inspired when employing Code2City$_{\rm VR}$ to perform program comprehension tasks. Although our results cannot be considered conclusive, they justify further studies on the use of VR in software visualization and its effect on feelings and emotions.

As for the users' thinking, Code2City$_{\rm VR}$ is comparable with Code2City and Eclipse. Regardless of the tool, the thinking is generally positive. Indeed, a small positive effect in favor the city metaphor implementation (*i.e.,* Code2City and Code2City$_{\rm VR}$) is observed with respect to Eclipse (*e.g.,* the Code2City$_{\rm VR}$'s users are significantly more stimulated to explore the program than those using Eclipse).

Concluding, we can positively answer our RQ as follows: the use of a VR-based implementation of the city metaphor affects users feelings and emotions, while the thinking about this implementation is positive and comparable with that of a traditional implementation of that metaphor and it is slightly better than the thinking about a non-visual exploration tool like Eclipse.

5.5 Threats to Validity

We discuss the threats the could affect the validity of our results with respect to internal, external, construct, and conclusion validity.

Internal Validity concerns uncontrolled factors that may alter the effect of the treatments on the dependent variables:

– *Selection.* Allowing volunteers to take part in a study may influence the results. In fact, volunteers might be more motivated than the whole population [41].
– *Resentful demoralization.* A participant using Eclipse could be less motivated, so she might not behave as good as she generally does.

External Validity regards the ability to generalize the results:

– *Interaction of selection and treatment.* Students might not be representative as professionals, thus the generalizability of the results might be questionable. We believe the use of students in our study is appropriate as the literature suggests [6]. In addition, we adequately trained the participants in performing the program comprehension tasks so making them experts in the use of the considered tools (see Section 2.3).

Construct Validity concerns the relation between theory and observation:

- *Hypothesis guessing.* The participants might guess the study goals and thus behave on the basis of their guesses.
- *Evaluation apprehension.* Some participants might be afraid of being evaluated. To mitigate this kind of threat, we informed the participants that their data would be treated anonymously.

Conclusion Validity regards treatments versus outcomes:

- *Hypotheses guessing.* The participants were aware of being part of a study regarding the use of software visualization tools. To mitigate this kind of threat, we did not provide any information about the actual study goal.
- *Reliability of measures.* This kind of threat depends on the measures used to assess feelings, emotions, and thinking. To deal with the reliability of measure, we used standard approaches to assess feelings, emotions, and thinking (*e.g.,* PANAS questionnaire).

6 Conclusion and Future Work

We presented the results of an empirical evaluation to study the city metaphor implemented in a 3D-based tool as well as in a VR-based tool. We contrasted these tools with a popular IDE (*i.e.,* Eclipse) with respect to users' feelings, emotions, and thinking, while performing program comprehension tasks on Java source code. The most important takeaway results of our investigation can be summarized as follows. First, the use of a VR-based implementation of the city metaphor significantly affects users' feelings and emotions. Second, the users' thinking about this implementation is positive and comparable with that of a traditional 3D implementation of this metaphor and it is slightly better than the thinking about Eclipse.

As future work, we plan to conduct replications of our study where we are going to focus on applications with defects. For example, this would help to increase our body of knowledge on the application of Virtual Reality (VR) to deal with defective applications. Finally, it is of paramount importance before adopting a new technique to assess the trade-off between advantages and disadvantages. As for VR, we need specialized hardware. Another disadvantage is that VR might produce physical discomforts. Therefore, a possible question to answer could be: are all these disadvantages adequately paid back in terms of improved feelings, emotions, and thinking? Although our study goes in this direction, we cannot answer that question. To this end, we have planned a long-run investigation specifically conceived to study possible trade-offs between advantages and disadvantages in the adoption of VR in the software industry.

References

1. Bacchelli, A., Rigotti, F., Hattori, L., Lanza, M.: Manhattan-3d city visualizations in eclipse. In: Proceedings of ECLIPSE IT, pp. 307–310 (2011)

2. Balogh, G., Beszédes, Á.: Codemetropolis - code visualisation in minecraft. In: Proceedings of International Working Conference on Source Code Analysis and Manipulation, pp. 136–141. IEEE CS Press (2013). DOI 10.1109/SCAM.2013.6648194
3. Bartlett, M.S.: Properties of sufficiency and statistical tests. Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences **160**(901), 268–282 (1937). DOI 10.1098/rspa.1937.0109
4. Canfora, G., Di Penta, M.: New frontiers of reverse engineering. In: Workshop on the Future of Software Engineering, May 23-25, 2007, Minneapolis, MN, USA, pp. 326–341 (2007)
5. Capece, N., Erra, U., Romano, S., Scanniello, G.: Visualising a software system as a city through virtual reality. In: AVR (2), *Lecture Notes in Computer Science*, vol. 10325, pp. 319–327. Springer (2017)
6. Carver, J., Jaccheri, L., Morasca, S., Shull, F.: Issues in using students in empirical studies in software engineering education. In: Proceedings of International Symposium on Software Metrics, METRICS '03, pp. 239–. IEEE, Washington, DC, USA (2003)
7. Dunn, O.J.: Multiple comparisons among means. Journal of the American Statistical Association **56**(293), 52–64 (1961). DOI 10.1080/01621459.1961.10482090
8. Dunn, O.J.: Multiple comparisons using rank sums. Technometrics **6**(3), 241–252 (1964)
9. Fittkau, F., Krause, A., Hasselbring, W.: Exploring software cities in virtual reality. In: Proceedings of Working Conference on Software Visualization, pp. 130–134 (2015). DOI 10.1109/VISSOFT.2015.7332423
10. Fittkau, F., Krause, A., Hasselbring, W.: Software landscape and application visualization for system comprehension with explorviz. Information and Software Technology (2016)
11. Francese, R., Risi, M., Scanniello, G., Tortora, G.: Users' perception on the use of metricattitude to perform source code comprehension tasks: A focus group study. In: Proceedings of International Conference Information Visualisation, pp. 8–13. IEEE CS Press (2017). DOI 10.1109/iV.2017.26
12. Fucci, D., Scanniello, G., Romano, S., Juristo, N.: Need for sleep: the impact of a night of sleep deprivation on novice developers' performance. IEEE Transactions on Software Engineering pp. 1–1 (2018). DOI 10.1109/TSE.2018.2834900
13. Graziotin, D., Wang, X., Abrahamsson, P.: Software developers, moods, emotions, and performance. IEEE Software **31**(4), 24–27 (2014). DOI 10.1109/MS.2014.94
14. Graziotin, D., Wang, X., Abrahamsson, P.: The affect of software developers: Common misconceptions and measurements. In: Proceedings of International Workshop on Cooperative and Human Aspects of Software Engineering, pp. 123–124 (2015). DOI 10.1109/CHASE.2015.23
15. Kapec, P., Brndiarov, G., Gloger, M., Mark, J.: Visual analysis of software systems in virtual and augmented reality. In: International Conference on Intelligent Engineering Systems, pp. 307–312 (2015). DOI 10.1109/INES.2015.7329727
16. Koschke, R.: Software visualization in software maintenance, reverse engineering, and reengineering: a research survey. Journal of Software Maintenance **15**(2), 87–109 (2003)
17. Kuutila, M., Mäntylä, M.V., Claes, M., Elovainio, M.: Daily questionnaire to assess self-reported well-being during a software development project. In: Proceedings of International Workshop on Emotion Awareness in Software Engineering, SEmotion '18, pp. 39–43. ACM (2018). DOI 10.1145/3194932.3194942
18. Lanza, M., Ducasse, S.: Polymetric Views-A Lightweight Visual Approach to Reverse Engineering. IEEE Transaction on Software Engineering **29**(9), 782–795 (2003)
19. Maletic, J.I., Leigh, J., Marcus, A., Dunlap, G.: Visualizing object-oriented software in virtual reality. In: Proceedings of International Workshop on Program Comprehension, pp. 26–35 (2001). DOI 10.1109/WPC.2001.921711
20. Mäntylä, M.V., Novielli, N., Lanubile, F., Claes, M., Kuutila, M.: Bootstrapping a lexicon for emotional arousal in software engineering. In: Proceedings of International Conference on Mining Software Repositories, MSR '17, pp. 198–202. IEEE Press (2017). DOI 10.1109/MSR.2017.47
21. Marcus, A., Comorski, D., Sergeyev, A.: Supporting the evolution of a software visualization tool through usability studies. In: Proceedings of the International Workshop on Program Comprehension, pp. 307–316. IEEE Computer Society, Washington, DC, USA (2005)
22. Merino, L., Bergel, A., Nierstrasz, O.: Overcoming issues of 3d software visualization through immersive augmented reality. In: Proceedings of Working Conference on Software Visualization, pp. 54–64 (2018)

23. Merino, L., Ghafari, M., Anslow, C., Nierstrasz, O.: Cityvr: Gameful software visualization. In: Proceedings of International Conference on Software Maintenance and Evolution, vol. 00, pp. 633–637 (2017). DOI 10.1109/ICSME.2017.70
24. Müller, S.C., Fritz, T.: Stuck and frustrated or in flow and happy: Sensing developers' emotions and progress. In: Proceedings of International Conference on Software Engineering, ICSE '15, pp. 688–699. IEEE Press (2015). URL http://dl.acm.org/citation.cfm?id=2818754.2818838
25. Murgia, A., Tourani, P., Adams, B., Ortu, M.: Do developers feel emotions? an exploratory analysis of emotions in software artifacts. In: Proceedings of Working Conference on Mining Software Repositories, MSR 2014, pp. 262–271. ACM (2014). DOI 10.1145/2597073.2597086
26. Romano, S., Capece, N., Erra, U., Scanniello, G., Lanza, M.: On the use of virtual reality in software visualization: The case of the city metaphor. Information & Software Technology (under review). URL www2.unibas.it/gscanniello/CityMetaphor/IST2018.pdf
27. Romano, S., Scanniello, G., Fucci, D., Juristo, N., Turhan, B.: The effect of noise on software engineers' performance. In: Proceedings of International Symposium on Empirical Software Engineering and Measurement, ESEM '18, pp. 9:1–9:10. ACM (2018). DOI 10.1145/3239235.3240496
28. Rudel, M., Ganser, J., Koschke, R.: A controlled experiment on spatial orientation in vr-based software cities. In: Proceedings of Working Conference on Software Visualization, pp. 21–31 (2018)
29. Shapiro, S.S., Wilk, M.B.: An analysis of variance test for normality (complete samples). Biometrika **52**(3/4), 591–611 (1965)
30. Souza, R., Silva, B., Mendes, T., Manoel, M.: Skyscrapar: An augmented reality visualization for software evolution. In: Brazilian Workshop on Software Visualization, pp. 17–24 (2012)
31. Storey, M.A.D., Wong, K., Muller, H.A.: How do program understanding tools affect how programmers understand programs. In: Proceedings of the Working Conference on Reverse Engineering, pp. 12–. IEEE Computer Society, Washington, DC, USA (1997)
32. Teyseyre, A.R., Campo, M.R.: An overview of 3d software visualization. IEEE Trans. Vis. Comput. Graph. **15**(1), 87–105 (2009)
33. Tony, A., Seewon, R., Ingoo, H.: The impact of web quality and playfulness on user acceptance of online retailing. Information & Management **44**(3), 263 – 275 (2007)
34. Tukey, J.W.: Comparing individual means in the analysis of variance. Biometrics **5**(2), 99–114 (1949)
35. Vegas, S., Apa, C., Juristo, N.: Crossover designs in software engineering experiments: Benefits and perils. IEEE Transactions on Software Engineering **42**(2), 120–135 (2016)
36. Watson, D., Clark, L.A., Tellegen, A.: Development and validation of brief measures of positive and negative affect: the panas scales. Journal of personality and social psychology **54**(6), 1063–1070 (1988)
37. Wettel, R., Lanza, M.: Program comprehension through software habitability. In: Proceedings of International Conference on Program Comprehension, pp. 231–240 (2007). DOI 10.1109/ICPC.2007.30
38. Wettel, R., Lanza, M.: Visual exploration of large-scale system evolution. In: Proceedings of Working Conference on Reverse Engineering, pp. 219–228. IEEE Computer Society (2008)
39. Wettel, R., Lanza, M.: Visually localizing design problems with disharmony maps. In: Proceedings of International Symposium on Software Visualization, pp. 155–164. ACM (2008)
40. Wettel, R., Lanza, M., Robbes, R.: Software systems as cities: A controlled experiment. In: Proceedings of International Conference on Software Engineering, pp. 551–560. ACM (2011). DOI 10.1145/1985793.1985868
41. Wohlin, C., Runeson, P., Hst, M., Ohlsson, M.C., Regnell, B., Wessln, A.: Experimentation in Software Engineering. Springer Publishing Company, Incorporated (2012)

A - Comprehension Tasks

In this appendix, we report the comprehension tasks the participants had to carry out as well as the rationale behind them and the corresponding concerns (see Table 14). The description of these tasks is taken from the paper by Wettel *et al.* [40].

Table 14 Description of the comprehension tasks taken from the paper by Wettel *et al.* [40].

Tasks
A1. Locate all the unit tests of the program and identify the convention (or lack thereof) used by the developers to organize the tests. *Rationale.* Test classes are typically defined in packages according to a project-specific convention. Before integrating their work in the system, developers need to understand how the test classes are organized. Software architects design the high-level structure of the system (which may include the convention by which test classes are organized), while quality assurance engineers monitor the consistency of applying these rules in the system. *Concern.* Structural understanding.
A2.1 Look for the term *T1* in the names of types and their fields and methods, and describe the spread of these types in the system. *Rational.* Assessing how domain knowledge is encapsulated in source code is important in several scenarios. To understand a system they are not familiar with, developers often start by locating familiar domain concepts in the source code. Maintainers use concept location on terms extracted from change requests to identify where changes need to be performed in the system. Software architects want to maintain a consistent mapping between the static structure and domain knowledge. Each of these tasks starts with locating a term or set of terms in the system and assess its dispersion. *Concern.* Concept location.
A2.2 Look for the term *T2* in the names of types and their fields and methods, and describe the spread of these types in the system. *Rationale.* Same as for task A2.1 (but the spread of the term *T2* is different). *Concern.* Concept location.
A3. Evaluate the change impact of the class *C* by considering its caller types. The assessment is done in terms of both intensity (number of potentially affected types) and dispersion (how these types are distributed in the program). *Rationale.* Impact analysis allows one to estimate how a change to a part of the system impacts the rest of the system. Although extensively used in maintenance activities, impact analysis may also be performed by developers when estimating the effort needed to perform a change. It also gives an idea of the quality of the system: A part of the system which requires a large effort to change may be a good candidate for refactoring. *Concern.* Change impact analysis.
A4.1 Find the 3 types with the highest number of methods. *Rationale.* Classes in object-oriented systems ideally encapsulate one single responsibility. Since methods are the classs unit of functionality, the number of methods metric is a measure of the amount of functionality of a class. Classes with an exceptionally large number of methods make good candidates for refactoring (*e.g.,* split class) and, therefore, are of interest to practitioners involved in either maintenance activities or quality assurance. *Concern.* Metric-based analysis.
B1.1. Identify the package with the highest percentage of god classes. *Rationale.* God classes are classes that tend to incorporate an overly large amount of intelligence. Their size and complexity often make them a maintainers nightmare. Keeping these potentially problematic classes under control is important. By maintaining the ratio of god classes in packages to a minimum, the quality assurance engineer keeps this problem manageable. For a project manager, in the context of the software process, packages represent work units assigned to the developers. Assessing the magnitude of this problem allows him to take informed decisions in assigning resources. *Concern.* Focused design assessment.
B1.2. Identify the god class containing the largest number of methods. *Rationale.* It is difficult to prioritize candidates for refactoring from a list of god classes. In the absence of other criteria, the number of methods can be used as a measure of the amount of functionality for solving this problem related to maintenance and quality assurance. *Concern.* Focused design assessment.
B2.1. Identify the dominant class-level design problem in the program. *Rationale.* God class is only one of the design problems that can affect a class. A similar design problem is the brain class, which accumulates an excessive amount of intelligence, usually in the form of brain methods. Finally, data classes are just “dumb” data holders without complex functionality, but with other classes strongly relying on them. Gaining a “big pictur” of the design problems in the system would benefit maintainers, quality assurance engineers, and project managers. *Concern.* Holistic design assessment.

Table 15 Some descriptive statistics for PAS and NAS grouped by tool and kind of participants.

	Statistic	All			Undergraduates		
		Code2City	Code2City$_{\mathrm{VR}}$	Eclipse	Code2City	Code2City$_{\mathrm{VR}}$	Eclipse
PAS	Mean	28.167	31.412	24.231	30.667	30.857	25.091
	Median	27	32	21	32	30	25
	SD	9.713	6.032	6.496	9.539	6.503	6.700
NAS	Mean	11.667	13.235	13.538	10.222	12.857	13.182
	Median	10	12	11	10	12	11
	SD	4.271	3.784	5.425	0.667	3.676	5.326

B - Descriptive Statistics

In this appendix, we provide the descriptive statistics values—mean, median, and SD (Standard Deviation). In particular, the descriptive statistics for the PAS and NAS values are shown in Table 15. These values are grouped by tool and kind of participants (*i.e.,* all the participants or only the undergraduates). We do not report the descriptive statistics for the graduates because of their low number (three in the Code2City and Code2City$_{\mathrm{VR}}$ groups, respectively, while two in the Eclipse group). When comparing the results of all the participants with those of the undergraduates only, we can notice that there are not huge differences in terms of either PAS or NAS values. In particular, for Code2City and Eclipse the PAS values are, on average, slightly better when considering the undergraduates only. As for Code2City$_{\mathrm{VR}}$, the PAS values are, on average, slightly better when considering all the participants. The NAS values are, whatever the tool was, slightly better when considering the undergraduates only. Summing up, we can observe no noticeable difference in the participants' emotions and feelings when comparing all of them with the undergraduates only.

Similarly to Table 15, we show the descriptive statistics for TQ, IQ, SQ, P, PEU, PU, AU, and BIU in Table 16. We can observe that their distributions are not so different when considering either all the participants or the undergraduates only. In particular, for Code2City and Eclipse the TQ values are, on average, slightly better when considering the undergraduates only. As for Code2City, the values are slightly better when considering all the participants. A similar trend can be observed for IQ, SQ, P, PU, AU, and BIU. With respect to PEU, for both Code2City$_{\mathrm{VR}}$ and Eclipse groups the mean value is slightly better when considering all the participants (while the mean PEU value is slightly better for Code2City when considering the undergraduates only). Again, we can conclude that there is no noticeable difference in the thinking of the undergraduates with respect that of all the participants.

Table 16 Some descriptive statistics for TQ, IQ, SQ, P, PEU, PU, AU, and BIU grouped by tool and kind of participants.

	Statistic	All			Undergraduates		
		Code2City	Code2City$_{VR}$	Eclipse	Code2City	Code2City$_{VR}$	Eclipse
TQ	Mean	27	26.471	25.385	28.889	25.714	25.909
	Median	28.5	26	26	30	26	26
	SD	6.954	3.573	2.815	3.756	3.124	2.256
IQ	Mean	35.833	35.294	34.154	36.889	34.786	35
	Median	36.5	36	35	37	36	36
	SD	5.508	4.356	4.758	3.1	4.423	4.427
SQ	Mean	10.75	11	11	12	10.786	11.182
	Median	11.5	11	11	12	11	12
	SD	3.494	1.696	2.449	1.414	1.672	2.639
P	Mean	32.417	30.706	25.308	33.444	30	26.091
	Median	33	31	28	31	30.5	28
	SD	8.49	5.072	7.387	4.902	4.977	6.7
PEU	Mean	39.417	39.529	38.692	40.444	39.286	37.455
	Median	39	42	40	43	41	38
	SD	6.082	4.888	5.879	6.366	4.615	5.52
PU	Mean	22.333	24.471	21.923	22.778	24.214	22
	Median	24	24	23	24	24	23
	SD	4.075	2.452	4.786	3.768	2.455	5.119
AU	Mean	30.667	31.118	29	32.444	30.786	29.364
	Median	33.5	31	30	34	30.5	30
	SD	6.513	2.848	4.378	3.127	2.607	4.501
BIU	Mean	16.167	17.588	15.692	17	17.286	15.818
	Median	17.5	18	16	18	16.5	16
	SD	5.149	2.451	3.172	3.317	2.525	3.188