The Journal of Systems and Software 210 (2024) 111985

Contents lists available at ScienceDirect

The Journal of Systems & Software

journal homepage: www.elsevier.com/locate/jss

The influence of the city metaphor and its derivates in software visualization☆

David Moreno-Lumbreras [a,*], Jesus M. Gonzalez-Barahona [a], Gregorio Robles [a], Valerio Cosentino [b]

[a] EIF@ Universidad Rey Juan Carlos, Fuenlabrada, Spain
[b] Eventbrite, Madrid, Spain

ARTICLE INFO

Keywords:
Software visualization
City metaphor
Software comprehension
Systematic mapping study
Visualizations
State of the art
Extended reality

ABSTRACT

Context: The city metaphor is widely used in software visualization to represent complex systems as buildings and structures, providing an intuitive way for developers to understand software components. Various software visualization tools have utilized this approach.
Objective: Identify the influence of the city metaphor on software visualization research, determine its state-of-the-art status, and identify derived tools and their main characteristics.
Method: Conduct a systematic mapping study of 406 publications that reference the first paper on the use of the city metaphor in software visualization and/or the main paper of the *CodeCity* tool. Analyze the 168 publications from which valuable information could be extracted, and build a complete categoric analysis.
Results: The field has grown considerably, with an increasing number of publications since 2001, and a changing research community with evolving interconnections between groups. Researchers have developed more tools that support the city metaphor, but less than 50% of the tools were referenced in their papers. Moreover, 85% of the tools did not use extended reality environments, indicating an opportunity for further exploration.
Conclusion: The study demonstrates the active and continually growing presence of the city metaphor in research and its impact on software visualization and its derivatives.

Editor's note: Open Science material was validated by the Journal of Systems and Software Open Science Board.

1. Introduction

The city metaphor is a popular approach in software visualization for representing complex software systems. By using the analogy of a city, with software components represented as buildings, quarters, and other structures, the city metaphor provides a simple and intuitive way for developers to understand the structure and relationships of software components. This approach is highly flexible and has been used in a variety of software visualization tools and applications, from the first approach – Software World (Knight and Munro, 1999) – to one of the most known approaches – *CodeCity* (Wettel and Lanza, 2007b). This metaphor is used in different topics, including code visualizers, software architecture visualizations, and performance analysis tools.

In the context of the continuous advancement and increasing accessibility of technology, this paper contributes to the argument that the development of new approaches utilizing the city metaphor in software visualization has become more feasible. This is expected to stimulate a surge in research and publications in this field. It is worth highlighting that research in the software visualization domain often demands substantial resources, encompassing the creation of innovative approaches and their subsequent evaluation. Nevertheless, the potential advantages offered by employing the city metaphor in software visualization, such as enhanced collaboration and communication, underline its significance as an area of interest for both researchers and practitioners.

One of the key benefits of the city metaphor is that it provides a high-level overview of a software system, making it easier for developers to identify potential problems and inefficiencies. For example, the city metaphor can help to visualize the flow of data and control between different software components, simplifying the identification of bottlenecks or potential performance issues (Ogami et al., 2017). Additionally, the city metaphor can be used to represent software systems at different levels of abstraction (Wettel and Lanza, 2008),

☆ Editor: Gabriele Bavota.

* Corresponding author.
E-mail address: david.morenolu@urjc.es (D. Moreno-Lumbreras).

https://doi.org/10.1016/j.jss.2024.111985
Received 26 April 2023; Received in revised form 11 January 2024; Accepted 23 January 2024
Available online 24 January 2024

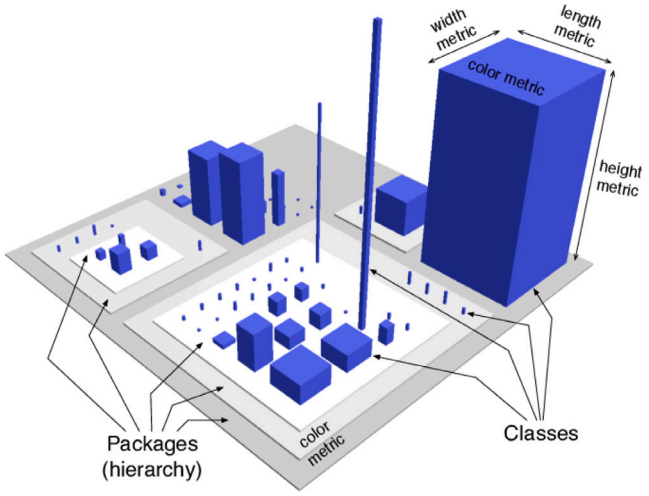

Fig. 1. Example of the use of the city metaphor in the *CodeCity* tool, showing the principles, artifacts definition and the metrics available for the buildings.
Source: Image from Wettel and Lanza (2008).

from the overall architecture of a system to the individual components and sub-components (Löwe et al., 2003), making it a useful tool for software developers working on different aspects of a project. In addition to the benefits in software comprehension, the city metaphor has the potential to improve collaboration and communication between software developers (Donalek et al., 2014; Hansen, 2021). By providing a clear and visual representation of a software system, the city metaphor makes it easier for team members to understand the overall structure and relationships between software components, facilitating better communication and collaboration in software projects.

Another important aspect of the city metaphor in software visualization is its ability to support software maintenance and evolution. By visualizing the structure and relationships of software components, the city metaphor can help developers identify potential problems and inefficiencies early on in the software development process, making the software system easier to maintain (Koschke, 2003; Erra and Scanniello, 2012) and evolve over time (Steinbrückner and Lewerentz, 2010a; Wettel and Lanza, 2008; Steinbrückner and Lewerentz, 2010b). This is particularly important for large-scale and complex software systems, which can be difficult to maintain and evolve without a clear understanding of the relationships between components.

However, the city metaphor is not without its limitations. One of the biggest challenges is the difficulty of accurately representing complex and dynamic software systems in a static visualization. For example, changes in the structure or behavior of a software system may not be immediately reflected in a city metaphor visualization, making it difficult for developers to understand the impact of these changes on the overall system. Additionally, the city metaphor may not be well-suited for representing certain types of software systems, such as distributed or parallel systems, where the relationships between components are highly complex and dynamic. Moreover, one of the popular layouts is the treemap, shown in Fig. 1. However adding a new element inside a treemap layout in software visualization can be difficult and disruptive. It often requires adjusting existing elements, leading to a cluttered layout. The process of adding a new element can also be time-consuming, as the entire treemap must be re-calculated. This can make it difficult to quickly add new elements to the treemap, affecting its flexibility and adaptability. As a consequence, the research community extended the metaphors used and studied variations of the layout, such as the work presented by Scheibel et al. (2018) presenting a treemap algorithm for evolving tree data and later presenting a survey of various treemap algorithms (Scheibel et al., 2020).

Despite these limitations, we argue that the city metaphor is a valuable software visualization tool for software developers, providing a simple and intuitive way to understand and visualize complex software systems. With its potential to improve collaboration and communication, support software maintenance and evolution, and provide a high-level overview of software systems, the city metaphor has potential to play an important role in the future of software visualization. As software systems continue to grow in complexity and scale, the city metaphor could provide an important tool for developers to understand and optimize these systems, making it easier to develop and maintain high-quality software.

This paper presents a systematic mapping study (SMS) of the use of the city metaphor in software visualization. Systematic mapping studies are crucial because they provide a comprehensive and structured overview of the existing literature, allowing researchers to identify gaps, trends, and opportunities for future research. By mapping the existing research on the city metaphor, we can better understand how this metaphor has been used in software visualization, what are the most common approaches, tools, and techniques, and what are the derivates and new implementations. Furthermore, a systematic mapping study can help to identify research questions that have not yet been fully addressed, providing a roadmap for future research in this field. With the growing interest in software visualization and the increasing complexity of software systems, the city metaphor is a promising approach that deserves further investigation and evaluation. Therefore, a systematic mapping study is a valuable tool for researchers, practitioners, and educators interested in understanding and using the city metaphor for software visualization.

The remainder of this paper is structured as follows. First of all, we describe the method used in the Systematic Mapping Study (SMS), with the research questions and the inclusion/exclusion criteria in Section 3. Section 4 describes how we extracted the data from the research literature, followed by the results from systematically analyzing 168 publications in Section 5. Related work is then presented in Section 2. We then discuss the implications in Section 6. The threats to the validity of our study can be found in Section 7. Finally, we draw conclusions and point out future research in Section 8.

2. Background and related work

In this section, we describe the related research, regarding the use of the city metaphor in the software visualization field and the SMSs and Systematic Literature Reviews (SLRs) studies that have been presented about software visualization.

City metaphor in software visualization

Metaphors play a critical role in software visualization, as they provide a systematic and stable relationship between two conceptual domains, as explained by the cognitive linguistics theory developed by Lakoff and Johnson (1980). However, the metaphor chosen for software visualization depends on the software artifacts being represented (Averbukh, 2001) and must be expressive enough to map relevant features. In the field of software visualization, there is a long history of using metaphors to represent software artifacts, and the development of 3D visualization technology has led to the creation of more realistic and easier-to-understand visual metaphors. For example, early 3D-oriented metaphors include the landscape metaphor (Balzer et al., 2004), which was used to visualize the structure of large software systems, and the solar system metaphor (Graham et al., 2004), which was used to visualize object-oriented software systems. In later years, *CodeTrees* was developed as a visualization tool for representing software as a collection of trees (Erra and Scanniello, 2012), and this concept was further extended in subsequent work (Maruyama et al., 2014). Overall, the use of metaphors in software visualization continues to be an important and active area of research, with new metaphors

and visualization techniques being developed to represent increasingly complex software systems.

The city metaphor has been utilized in various approaches for software visualization, with *Software World* (Knight and Munro, 1999) being the first implementation that visualized software systems as buildings in a city. Since then, numerous techniques have been explored to assist developers in maintaining software systems and supporting program comprehension tasks.

For instance, Panas et al. (2003, 2005, 2007) introduced a software city visualization that provides information about static and dynamic data, while Marcus et al. (2003) developed a city-like software visualization. Additionally, *Verso* (Langelier et al., 2005) was based on landscapes, but with an influence from the city metaphor. Overall, the city metaphor has been widely adopted in software visualization, and its use has led to the development of various techniques that aid developers in understanding software systems. In 2007 *CodeCity* was presented (Wettel and Lanza, 2007a), raising the approach to a new level implementation-wise. *CodeCity* showed that it could not only be used for program comprehension (Wettel and Lanza, 2007a), but also for software evolution analysis (Wettel and Lanza, 2008) and design problem analysis (Wettel and Lanza, 2008). It sparked a flood of tools and approaches building on the same metaphor, leading to slightly different visualizations, showing the power and flexibility of the metaphor. Scarsbrook et al. (2018) presented a tool for visualizing and debugging a large-scale JavaScript program structure with treemaps, and Brito et al. (2019) presented a similar approach focusing on the Go programming language. Steinbrückner and Lewerentz (2010b) proposed a different layout for the city, based on streets and sub-streets for the tree structure, allowing to observe the time evolution of the software system. Gamification has also been used in combination with the city metaphor to perform software comprehension tasks in *CodeMetropolis* (Balogh et al., 2016), based on the Minecraft game engine. *M3tricity* (Pfahler et al., 2020), a recent re-implementation of *CodeCity* by the original research group, is a web application to visualize software systems as evolving cities that treats evolution as a first-class concept.

Early explorations of using Virtual Reality (VR) for software visualization were conducted by Young and Munro (1998), who developed a technical implementation. Another early VR-based approach, *Imsovision* (Maletic et al., 2001), defined metrics for C++ and is still referenced in current literature. With recent advancements in technology, the use of VR for software visualization has become an active field of research. Fittkau et al. (2015) proposed a VR implementation of *ExplorViz* using the *WebVR* platform, focusing on object-oriented programming software systems' runtime and static characteristics. Vincur et al. (2017b,a) developed a VR city for analyzing object-oriented software. *Getaviz* (Baum et al., 2017) also uses the city metaphor to generate structural, behavioral, and evolutionary views of software systems for empirical evaluation. In contrast, *CityVR* (Merino et al., 2017), developed with *Unity3D*, provides similar metrics to the original *CodeCity* and includes interactions using the user's gaze and VR controllers. Another VR-based software visualization technique is the use of the city metaphor in the *Unreal Engine 4* by Capece et al. to visualize Java systems (Capece et al., 2017). Additionally, Misiak et al. (2018) proposed the island metaphor to visualize OSGi-based software systems and emphasize dependencies, while Schreiber and Brüggemann (2017) presented an interactive tool that visualizes OSGi-based systems with different components, packages, services, and dependencies in 3D using a box-based metaphor.

SMSs and SLRs in software visualization

In the software visualization field, several SMSs and SLRs have been conducted to analyze and synthesize the research on different topics related to software visualization.

One of the early works in this field is by Diehl (2007), who conducted a review of over 300 papers published in the software visualization field between 1990 and 2006, offering an overview of state-of-the-art visualizations at that time. Significant studies that have influenced our work include the one presented by Merino et al. (2018). Their systematic literature review (SLR) focused on papers from the SOFTVIS/VISSOFT conferences, revealing that 62% of proposed software visualization approaches lacked validation through empirical experiments or comparative analyses against other methods. Although closely related to our study in its focus on software visualizations, Merino et al.'s work concentrated solely on the SOFTVIS/VISSOFT conference, while ours delves into the use of the city metaphor and its derivatives, examining characteristics and approaches derived from these studies. Another SLR shaping our perspective is the one conducted by Mattila et al. (2016), which presented results from a systematic literature review spanning from 2012 to 2018, emphasizing the analysis of visualization aims. Mattila's findings highlighted software structure, behavior, and evolution as prominent topics, prompting our investigation into whether the city metaphor relates to these aspects. Seriai et al. (2014) conducted a systematic mapping study with a specific focus on validation methods in software visualization, examining over 700 articles. Their findings highlighted a deficiency in rigor when it comes to validating software visualization tools and techniques. In contrast, our study centered on the utilization of metaphors for visualization, excluding the specific analysis of whether the approaches have been validated. This distinction underscores our emphasis on exploring the diverse applications of metaphors in software visualization, contributing a unique perspective to the literature without delving into the validation aspects explored by Seriai et al.. Furthermore, Novais et al. (2013) conducted a systematic mapping study with a specific emphasis on software evolution visualization, addressing gaps in studies related to goals, strategies, and approaches in this context. While Novais et al.'s work delves into the broader spectrum of software evolution, our study distinguishes itself by focusing specifically on the use of the city metaphor and its derivatives in representing software. Our primary emphasis lies in examining the various ways in which the city metaphor is employed for software visualization, without restricting our scope to a particular use case such as software evolution.

Apart from those publications, there are other surveys and studies on this topic. Caserta and Zendra conducted a comprehensive survey to visualize the static aspects and evolution of software (Caserta and Zendra, 2010). Carpendale and Ghanam reviewed the literature on software architecture visualization (Carpendale and Ghanam, 2008), while Basit et al. discussed code clone visualizations (Basit et al., 2015). Several meta-studies have also focused on software execution and performance visualization. Hamou-Lhadj and Lethbridge conducted a survey on the tools and techniques used for visualizing software execution traces (Hamou-Lhadj and Lethbridge, 2004), while Isaacs et al. presented a state-of-the-art report on performance visualization (Isaacs et al., 2014). Koschke conducted a meta-study on software visualization supporting software maintenance (Koschke, 2003), and Schots et al. discussed visualizations to support software reuse (Schots et al., 2014). In addition, Paredes et al. conducted a mapping study on software visualization tools used in agile software development teams (Paredes et al., 2014). Storey et al. presented a framework for classifying software visualization tools based on their intent, information, presentation, interaction, and effectiveness, which was developed through a survey of several software visualization tools (Storey et al., 2005). Bassil and Keller also presented a survey of software visualization tools, collecting users' perceptions on what worked and what did not work using a questionnaire (Bassil and Keller, 2001). Teyseyre and Campo presented an overview of 3D software visualizations, analyzing existing projects in terms of visual representation, interaction, evaluation, development tools, and their scope (Teyseyre and Campo, 2008).

Table 1
Number of studies citing the papers selected in the inclusion criteria by research database.

	Google Scholar	Semantic Scholar	IEEE Xplore
Software World	126	89	31
CodeCity	362	267	109

Table 2
Categories related to the basic information about the publications analyzed.

Acronym	Type	Values	Description
TITLE	String	–	Title of the study
AUTHORS	List	–	List of authors of the study
YEAR	Number	–	Year of publication of the study
INCLUDED	Boolean	True, False	Is the study included in the analysis?

3. Method

A systematic mapping study (SMS) is a type of research method used to identify, critically evaluate, and synthesize all relevant empirical evidence on a specific research question (Petersen et al., 2008). Its purpose is to provide a comprehensive overview of the current state of knowledge, identify gaps in the literature, and convey future research by highlighting areas of consensus, controversy, and emerging trends. By following a rigorous and systematic process, the study aims to reduce bias and ensure the results are reliable and robust. We follow the SMS guidelines proposed by Kitchenham et al. (2002) and Petersen et al. (2008), encompassing the following key principles: clearly defining research questions, specifying the search strategy, determining data extraction methods, elucidating the study selection process, articulating the data synthesis approach, outlining quality assessment considerations, explaining the mapping process, and defining the structure for reporting the results.

3.1. Research questions

This SMS aims to study and analyze the use and of the city metaphor and its derivates in the software visualization field, including papers and tools. Consequently, we formulated the following research questions:

RQ_1: Has the use of the city metaphor in software visualization increased?

Motivation. With the continuous advancement and increasing accessibility of technology, our main motivation is to prove that it is now easier to develop new approaches using the city metaphor in software visualization, and to understand the state of the art in this field. We hypothesize that the city metaphor is of great importance within the software visualization research field, and we aim to analyze the temporal evolution of the field, to observe if it is currently an active research field and if it is in growth or in decline, in terms of number of scientific publications and development of tools. Our ultimate goal is to gain insight into the potential benefits of using the city metaphor in software visualization, such as improved collaboration and communication, and to encourage further research and development in this area.

RQ_2: How does the number of research teams evolve over time?

Motivation: This research question aims to analyze the evolution of the software visualization research community in the field by examining scientific publications and collaborations among research groups. Specifically, we aim to investigate whether advancements in technology and increased accessibility have led to more collaboration among research groups, and whether new research groups have emerged in this field. Additionally, we seek to determine if established research groups continue to produce studies in this area of research.

RQ_3: Are the papers exploring improvements or extending the city metaphor?

Motivation: The motivation behind this research question lies in the desire to uncover whether researchers are actively enhancing the existing visualization techniques or venturing into uncharted territories by extending the city metaphor. By addressing this question, we can gain valuable insights into the innovative strategies employed in the field.

RQ_4: Are researchers developing more tools that support the city metaphor?

Motivation: The objective of this question is to investigate the software produced as a result of scientific publications in the field of software visualization, with a focus on analyzing the temporal evolution of the field. By examining important aspects such as the number of publications that use each software tool, we aim to determine if the research community has produced more or less software over time. This information will provide insights into the trends and developments in the field of software visualization, helping researchers and practitioners understand the state of the art and identify areas for future research. Given that the development of prototypes is closely linked to this research field, one of the most important factors is whether it is available for use and reproducing the scenarios proposed in the publication. With the advances in technology and greater accessibility of tools, it is possible that the number of developed tools may have increased, and that they may be more readily available for use and reproduction. This includes whether there is a link within the publication itself to access the software, whether this link is still up and can be downloaded on the day of the study, and whether the source code is still available.

RQ_5: Are the prototypes exploring new emerging immersive technologies?

Motivation. Technology is advancing rapidly, and new technologies such as Virtual Reality (VR), Augmented Reality (AR), and Extended Reality (XR) are emerging, which can have a significant impact on the use of the city metaphor. These emerging technologies offer exciting new possibilities for exploring and understanding the city metaphor in a 3D environment. As such, it is important to investigate which of these environments are best suited for the tools, as well as other configuration and feature considerations that may arise with these technologies.

3.2. Inclusion and exclusion criteria

Once defined the research questions, we present the inclusion and exclusion criteria for the SMS. In addition, we describe the search strategy used for primary studies and tools, the search source and the reasons for removing papers from the list.

The inclusion criteria address all published studies written in English that cite either (i) the publication that presents the first software visualization using the city metaphor, authored by Knight and Munro (1999), or (ii) the work presented by Wettel and Lanza (2007b), a work considered seminal in the field.

Before accepting a paper into the SMS, we excluded publications that are duplicates, i.e., a less matured version (conference, workshop, Ph.D. thesis...) of a matured version (usually a journal publication). In those cases, we only considered the matured version. When we found a short and a long version of the same publication, we have chosen the longer version. However, in those cases where the publication is a Ph.D. thesis and a related (peer-reviewed) publication exists in a workshop, conference, or journal, we have discarded the thesis in favor of the latter, because conference and journal publications are peer-reviewed and PhD theses are not. Documents that are a false positive (i.e., not a real scientific publication, internal report, etc.) have also been excluded.

Table 3
Categories related to the features of a publication.

Acronym	Type	Values	Description
LAYOUT	Boolean	True. False	Only True if the study clearly describes the algorithm – not just mentions it, uses it, as described somewhere else.
DOC	Options	Thesis, PhD, Journal, Conference, Workshop, Report, Other	Document type
NOVEL	Boolean	True, False	If the study presents an advance in the state of the art. For example, a new metaphor or extension, a new implementation, a new feature, or some new use case of the city metaphor.
METAPHOR	Boolean	True, False	Metaphor Novelty. If the paper presents a variation of the city metaphor presented in *CodeCity* (Wettel and Lanza, 2007b).
METAPHOR TYPE	String	CITY, ISLANDS, Other	Metaphor type used. Possible types are city (inspired by *CodeCity*), forests, islands, world (which includes a mix of the previous types), and constellation.
IMPL	Boolean	True, False	Implementation Novelty. Does the paper present a new implementation or new features in an implementation?
CASE	Boolean	True, False	Use Case Novelty. Does the paper present a new use case?

3.3. Search strategy for primary studies

First, we selected Google Scholar,[1] Semantic Scholar,[2] and IEEE Xplore[3] as the databases for searching for studies. We finally have looked exclusively at Google Scholar, since the number of studies retrieved there is significantly higher than from the other research databases. Studies like the one presented by Wolny et al. (2020), the one by O'Donovan et al. (2015), and the one by Rodrigues et al. (2019) relied solely on Google Scholar due to its superior search results. Table 1 shows the number of studies of the three databases when searching for publications that cite the two papers selected in the inclusion criteria. Google Scholar may return false positives, including non-peer-reviewed documents like slide sets and notes, alongside publications. However, for our specific research purposes, this aspect is not problematic as its results encompass a superset of information. We acknowledge that Google Scholar does return other types of documents, such as non-peer-reviewed materials; however, these were appropriately filtered out in subsequent phases. Some other databases, like Scopus, were excluded from consideration due to the limited number of publications they retrieved.

4. Study quality assessment

This section explains how we obtained the data to show an overall picture of the use of the city metaphor and its derivates in the software visualization research field.

4.1. Quality assessment criteria

Our approach for studying the quality assessment is based on Kitchenham et al. (2002) concept of quality. Thus, our assessment is focused on identifying only those studies that include factors related to software visualization using the city metaphor (or a derivate from it).

Phase 1: Validating the quality of the dataset

To validate the database of studies and its corresponding quality, we chose four studies (Caserta and Zendra, 2011; Pfahler et al., 2020; Weninger et al., 2020; Hoff et al., 2022) from this field of research, using the query *("codecity" OR "city metaphor" OR "code cit*") AND software AND (visualizat* OR "3d")* in the Google Scholar database, and we verified that the references of these articles were within the database that we have formed. These four studies are from different years, one is a journal publication, and the others are publications from different years at the VISSOFT conference,[4] dedicated exclusively to software visualization.

Phase 2: Extract publications only related to the topic

With this quality-validated database of studies, and including only the more mature ones if there are several, as described in the inclusion criteria, we performed a superficial analysis of each of the studies and discarded the studies that do not present 3D visualizations, except if they present a visualization which is based on the city metaphor or a new metaphor, or a clear evolution of it, in 2D. The final database after this process is composed of 168 studies.

4.2. Analysis of the categories to extract the studies

Before undertaking a comprehensive review of all 168 studies, the authors conducted a preliminary study to delineate the categories and information to be extracted. In this initial phase, two authors independently selected distinct subsets of studies, thoroughly analyzing them to identify pertinent categories for extraction. Following this, the two author proposed a set of categories based on their subset analysis. In a collaborative effort, all authors engaged in discussions to consolidate and refine the proposed categories. The aim was to ensure that each category had a defined and limited set of values. Through this iterative process of analysis and discussion, the final list of categories was collectively defined, laying the groundwork for the systematic categorization of all studies.

4.3. Categorizing information from studies

We have read and analyzed the 168 studies, extracted information from them, and divided them into categories. Table 2 describes the basic information about the publications analyzed, including the title, the author list, the year of the publication, and if the publication was included in our study. Table 3 depicts the categories related to the features that the publication presents. After that, we focus on the tools used or presented in the publication. Table 4 presents the name of the tool, if available, and information about the online availability of the tool including if it is still alive or if the source code is referenced in the publication. We also analyze if the tool is presented in a 2D conventional screen, and if the tool uses any kind of virtual or augmented reality for representing the visualization. Complementing this information, Table 5 represents the remaining information about the tool, such as if the tool is modular, if the tool includes some interaction with the user, or even if the tool can filter the data in some way. We also present information about the number of metrics available to use within the tool, and how many metrics the user can represent in the visualization simultaneously.

To categorize the studies, the initial data extraction was conducted by two authors who thoroughly reviewed all the studies and compiled pertinent information to populate the categories for each study. Following this, a validation step was introduced, involving a third author

[1] https://scholar.google.es/.
[2] https://www.semanticscholar.org/.
[3] https://ieeexplore.ieee.org/.
[4] https://vissoft.info/.

Table 4
Categories related to the availability of the tool that the publication uses/presents.

Acronym	Type	Values	Description
NAME	String	–	Name of the approach presented or used in the study.
DOMAIN	String	–	Domain of the study. In which domain could be classified the paper?
PUBIMPL	Boolean	True, False	Implementation publicly available. Does the paper include a clear reference (usually as a link, or as a part of the reproduction package) to the implementation? The reference may be to a service, to a binary, or to source code.
AVAILIMPL	Boolean	True, False	Implementation is available. Is the implementation publicly available at the time of analyzing the study?
SOURCE	Boolean	True, False	Source Code. Does the paper include a clear reference to public version of the source code?
SCREEN	Boolean	True, False	Screen as the display. Does the implementation presented in the paper show its results on a 2D screen?
SCENE	Options	VR, AR, XR, None (No AR or VR)	Does the implementation presented in the paper show its results in a 3D scene using Virtual/Augmented/Extended reality?

Table 5
Features and metrics information of the tool presented/used in the publication.

Acronym	Type	Values	Description
MODULAR	Boolean	True, False	True if the paper proposes a modular framework. For example, this property is true if the visualization relies on an intermediate representation (e.g., model) of the software system.
CONFIG	Boolean	True, False	Configurable. Is the implementation presented in the paper configurable, so that the user can specify different aspects of it before launching it? Examples of aspects to configuring: layout, metrics shown, aspect, and data to present.
INTERACT	Boolean	True, False	Interactive. Does the implementation presented in the paper allow for user interaction, via some user interface? Examples of user interaction: select a building to watch its metrics, change appearance, change scale.
NAV	Boolean	True, False	Navigation. Does the implementation presented in the paper allow the user to navigate through the approach?
FILTER	Boolean	True, False	Data Filtering. Does the implementation presented in the paper allow the user to filter the data interactively?
NOMET	Number	–	Total number of metrics that are available to represent, not those that are visualized at a given time. We could use ">n" in case it can present n metrics or more.
SIMULMET	Number	–	Simultaneous number of metrics available to represent in the visualization.

Table 6
Percentage and number of studies included after applying inclusion criteria.

Acronym	Description	Percentage	Number of studies
TRUE	Study included.	42.46	168
NO3DVIZ	No 3D visualizations in the study.	28.64	113
OTHER	Other reasons.	19.34	77
DUP	Duplicated.	7.28	29
NOT FOUND	Study not found.	2.26	9

who meticulously reviewed the information provided by the initial two authors. In instances where discrepancies or disagreements were identified, the third author performed a manual review of the relevant category and ascertained the accurate information. Subsequently, after validation by the third author, each original author was informed of the changes and provided with an opportunity to approve them.

5. Results

This section presents the results of our SMS. All the information, source code used for the analysis, figures, data, and details at the publication level and for each of the RQs, can be found in the online replication package, defined in Section 8.

After following the study inclusion criteria, Table 6 shows percentages and number of studies that have been included and excluded. The results and analysis in this section are done with the studies included after the inclusion/exclusion criteria described in Section 3.

We are going to divide this section into two subsections, one related to the analysis of the publications and the other related to the tools/prototypes used/presented.

Table 7
Percentage and number of the study type of those included after the inclusion criteria applied.

Publication type	Percentage	Number of studies
CONFERENCE	58.58	99
JOURNAL	13.61	23
THESIS	11.24	19
REPORT	7.69	13
PHD	4.73	8
OTHER	2.36	4
WORKSHOP	1.18	2
BOOK	0.59	1

5.1. Analysis of publications

Focusing on the type of publication, conference articles predominate in this topic, being the type of document with more than half of those included in our SMS. Many of these papers belong to the *VISSOFT* conference, as this conference has software visualization research as its aim. Table 7 represents the percentage of studies included and their type of publication in detail.

RQ_1: *Has the use of the city metaphor in software visualization increased?*

Our final sample included 168 papers. Given this sample, Fig. 2 represents how this number of publications is divided over the years by publication date. We can see that, as the years go by, the number of publications increases, having a clear turning point in 2007. This is due to the improvement of technologies and the greater accessibility of these to develop prototypes so more researchers are able to experiment with and develop software visualizations that use the city metaphor, increasing therefore the number of research teams. We can see that,

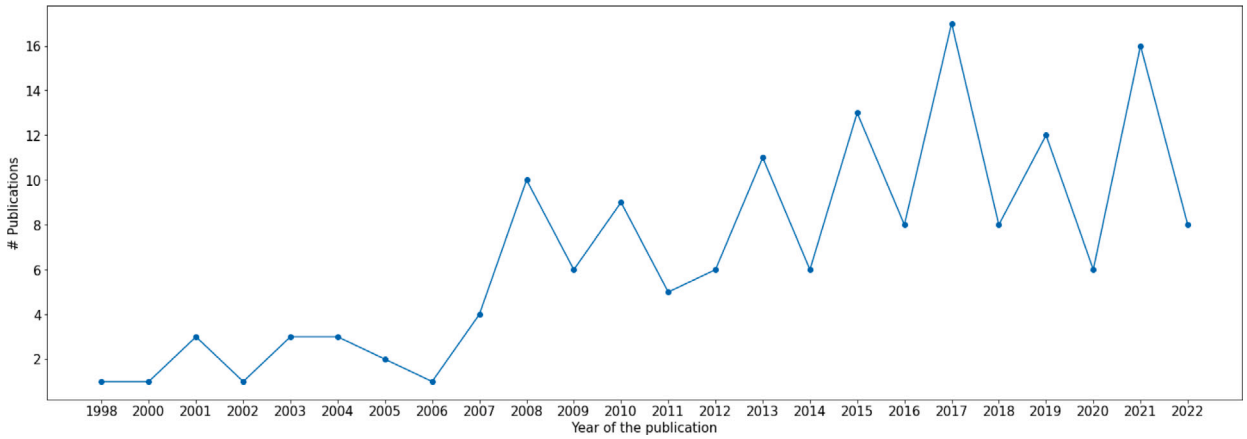

Fig. 2. Number of publications over time.

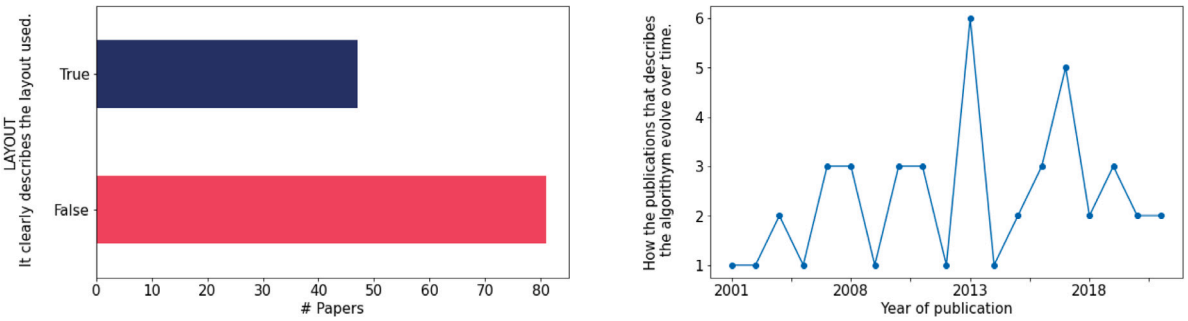

Fig. 3. Number of publications describing the placement algorithms (left) and how they are distributed over time (right).

Table 8
Authors with more than 3 publications, and their number of publications in the research topic.

Author	pubs
M Lanza	16
W Hasselbring	11
R Wettel	10
F Fittkau	7
U Erra	7
G Scanniello	7
R Minelli	6
JM Gonzalez-Barahona	5
R Koschke	5
A Krause	5
D Moreno-Lumbreras	5
C Zirkelbach	5
V Dashuber	4
G Langelier	4
M Philippsen	4
A Kuhn	4
O Nierstrasz	4
A Schreiber	4
N Capece	4
F Steinbrückner	4
M Misiak	4

within this topic, at least in the last 11 years, there have been always more than five publications per year, suggesting that this topic is stable and active in the research community.

If we follow Fig. 2 we can observe that the number of publications per year has been consistently above 5 since 2008, indicating a stable trend in terms of the number of publications.

One of the most important elements when using different metaphors to visualize software is the layout they use. Fig. 3 indicates that slightly less than half of the publications analyzed in this study provided a detailed explanation of the algorithm used for their software visualization. This suggests that some of these publications may have relied on pre-existing placement algorithms, rather than developing their own unique algorithm. According to the publication date, there has been at least one publication each year since 2001 that provides a detailed description of the algorithm. Furthermore, since 2014, there have been at least two publications each year that describe the algorithm. These findings suggest that research in this field is growing and that there is ongoing interest in developing and refining placement algorithms for software visualization.

Another indication of the growth of this field is the increasing number of different metaphors used to represent software, either as improvements on existing ones or completely new ones. As shown in Fig. 4, the number of unique metaphors used in this field has been steadily increasing over time. We can further support our hypothesis by observing that since 2010 at least two different metaphors based on the city metaphor have been used to visualize software.

And if we observe Fig. 5, we can see that the predominant metaphor used in software visualization is the city metaphor, the first one explored and on which most of the tools developed are based.

RQ_2: *How does the number of research teams evolve over time?*

The analysis of the articles in this field showed that there were a total of 265 unique authors. However, upon closer examination, it was found that more than 50% of these authors have only published one article.

The remaining authors who have published multiple articles were found to be primarily concentrated within a subcommunity of researchers. Table 8 lists the authors who have published more than three articles on the topic; many of these authors are co-authors on the same publications.

This observation is further supported by the network diagram shown in Fig. 6, which visualizes the interconnections between authors in

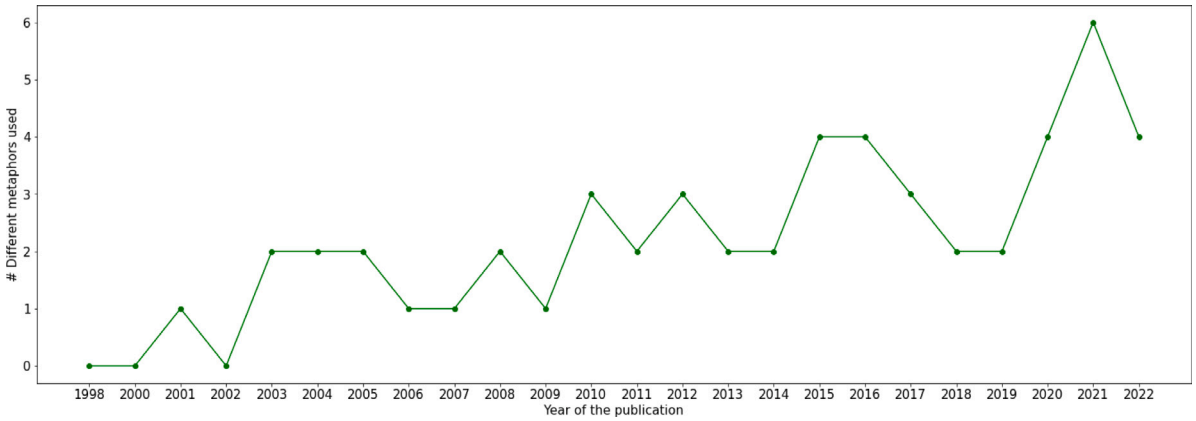

Fig. 4. Metaphors over time.

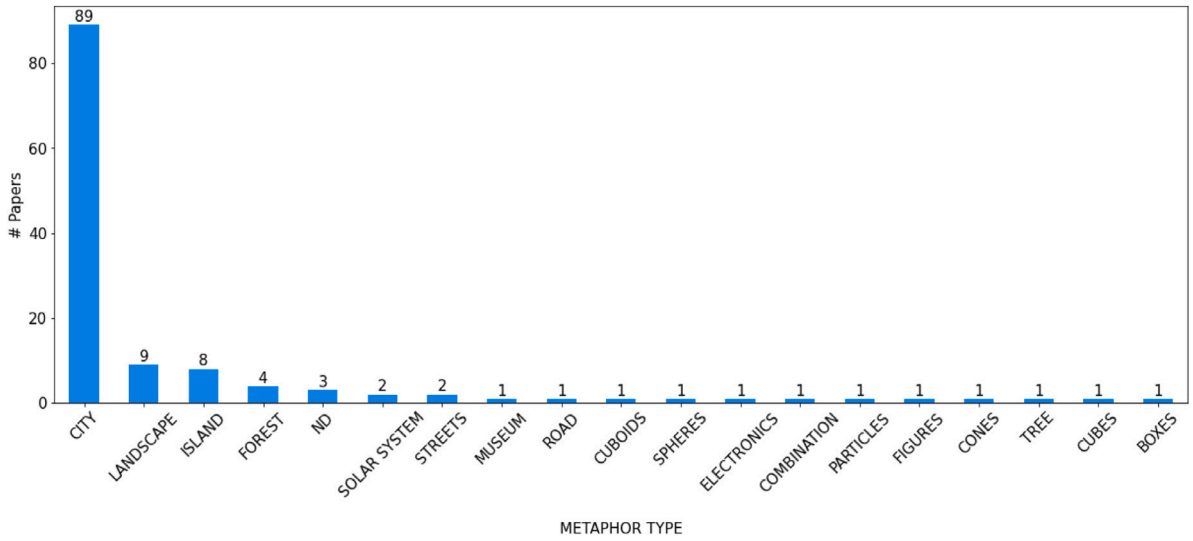

Fig. 5. Metaphors used in the publications.

the community. The diagram depicts each author as a node in the network, with relationships between authors represented by edges. The large nodes in the diagram correspond to the authors of codecity, a software visualization tool that has been a significant contribution to this field. Notably, the diagram shows that these authors have connections to other subcommunities within the larger community of researchers working on this topic.

One notable author is *M. Lanza*, who is considered one of the major exponents and still active researcher of this community. The diagram showing relationships between subcommunities indicates some level of collaboration and exchange of ideas among researchers from different groups within the field. However, the concentration of active researchers in a small subcommunity raises concerns about the diversity and representation within the broader community of researchers investigating the city metaphor in software visualization.

RQ_3: *Are the papers exploring improvements or extending the city metaphor?*

For this specific question, we analyzed the various metaphors used in the studies, as depicted in Fig. 5, along with four additional categories. The results are illustrated in Fig. 7.

- **Utilization of Different Metaphors**: Among the studies, 89 utilized the city metaphor, constituting more than 50% of all included studies. Other notable metaphors include the landscape metaphor, the island metaphor, and the forest metaphor, each employed by at least three studies. Additionally, there are over 10 different metaphors, each utilized by a single study, suggesting a need for further research to either validate or refine these less-explored metaphors.
- **Advancements in the State of the Art**: A significant majority of the studies (130 out of 168) made noteworthy contributions, advancing the state of the art in the field. These contributions involved innovative approaches utilizing new technologies such as VR or web technologies, techniques such as the algorithm selected for the layout, or concepts such as the use of the city for presenting data about software performance. This highlights the continuous efforts within the research community to enhance and broaden the application of the city metaphor in software visualization. Specific advancements made by these 130 studies will be explored in detail.
- **Variations of *CodeCity* Tool**: Among the studies that advanced the state of the art, a noteworthy subset of 92 focused on presenting variations of the *CodeCity* tool. *CodeCity*, as a cornerstone application in this domain, demonstrated its adaptability as researchers consistently explore ways to build upon and refine this influential tool. A variation of the *CodeCity* tool is defined as those studies that start with the idea of representing similar metrics/data but improve upon the original one.
- **New Implementations of City Metaphor Visualization**: Out of the studies that advanced the state of the art, 119 introduced new implementations of the city metaphor visualization.

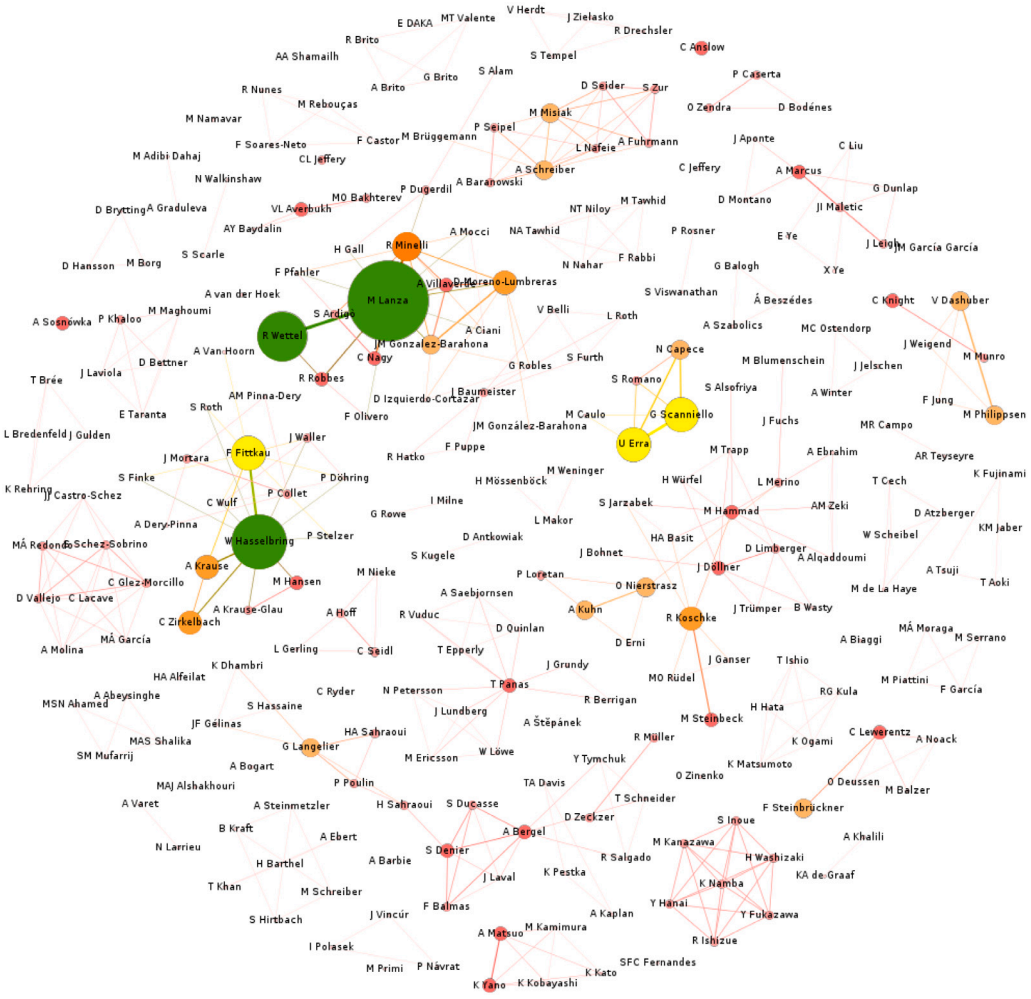

Fig. 6. Community network in the use of the city metaphor in software visualization.

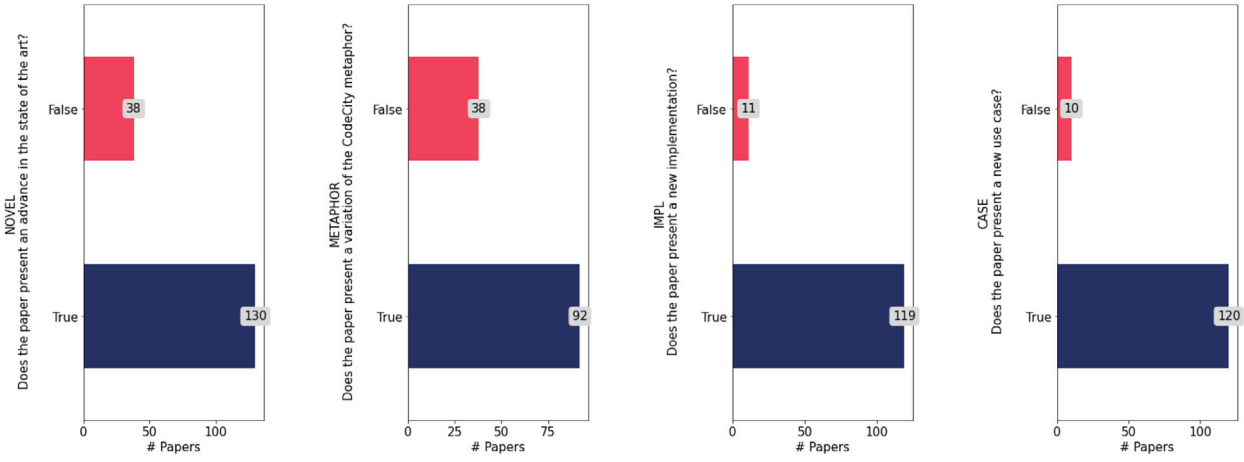

Fig. 7. Categories of papers included in the sample.

These implementations contribute to the diversity and versatility of techniques available for representing software systems, showcasing the dedication of researchers to explore innovative applications of the city metaphor.

- **Exploration of New Use Cases**: A substantial portion of the studies, specifically 120 out of the 130 that advanced the state of the art, delved into exploring new use cases for the city metaphor in software visualization. These use cases included the

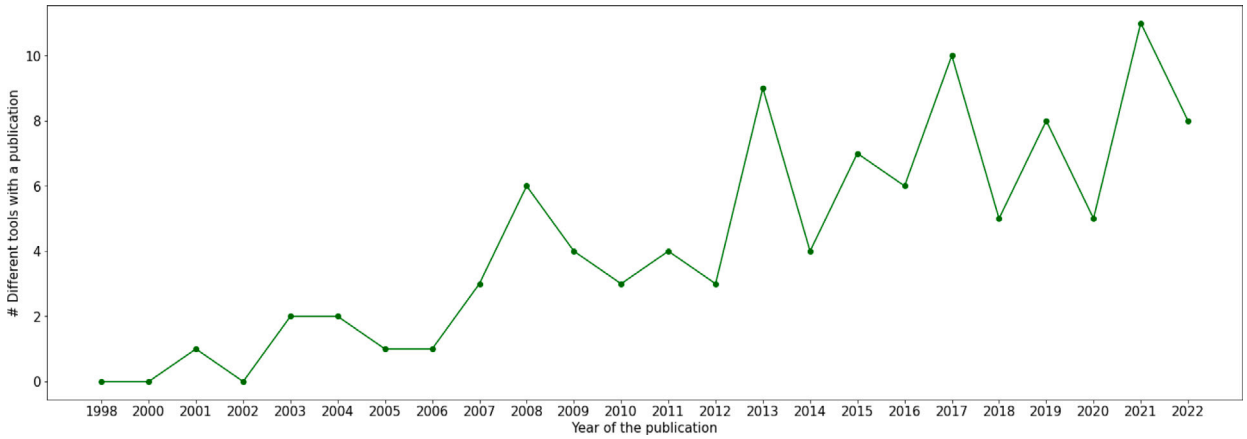

Fig. 8. Number of unique tools with a publication per year.

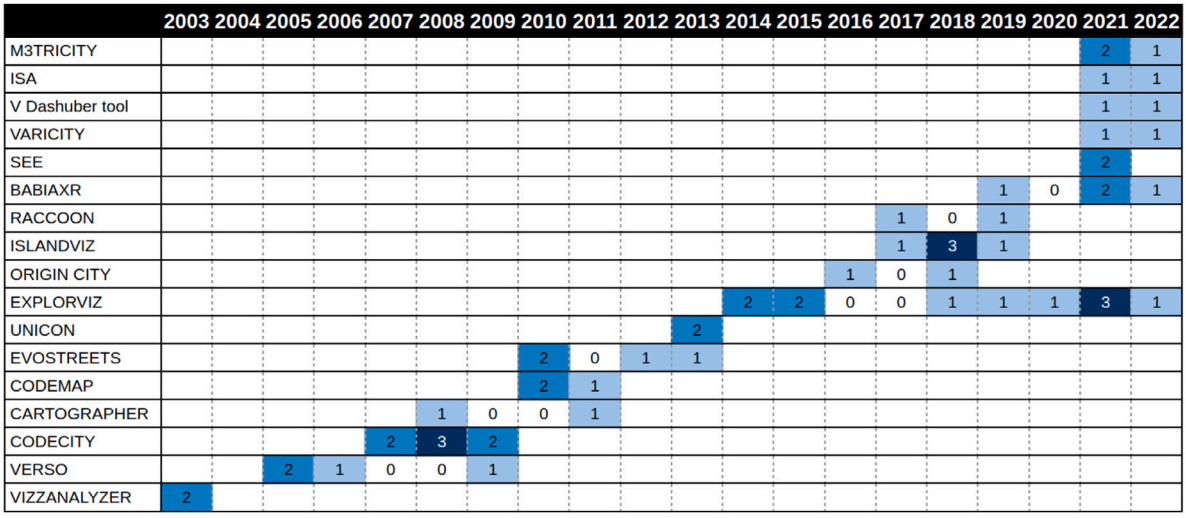

Fig. 9. Gantt diagram of the tools with more than one publication.

representation of different types of data and the exploration of diverse methods for visualizing additional metrics. This exploration underscores the adaptability of the city metaphor to various contexts and domains beyond its original scope, indicating an active extension of its application by researchers.

The comprehensive analysis of studies in the field of software visualization employing different metaphors revealed valuable insights. A majority of the studies, comprising over 50%, utilized the city metaphor, emphasizing its prevalent role. Noteworthy alternatives, such as the landscape, island, and forest metaphors, were employed by multiple studies, showcasing a diversified approach. The existence of over 10 different metaphors, each used by a single study, highlights areas requiring further research to validate or refine these less-explored metaphors. Additionally, a significant portion of the studies, around 76.3%, contributed to the state of the art by introducing innovative approaches, variations of the *CodeCity* tool, new implementations of city metaphor visualization, and exploration of new use cases. These findings underscore the dynamic nature of the research community, actively advancing and diversifying the application of the city metaphor in software visualization.

5.2. Analysis of tools

RQ$_4$: *Are researchers developing more tools that support the city metaphor?*

A total of 78 tools or approaches have been identified. But many of these tools have only been featured in one publication; only 17 of the tools appear in more than one publication. Given this, we consider these tools to be more mature than the rest. In order to investigate trends in the production of software tools in the field of software visualization, we analyzed the number of unique tools that were referenced in publications each year. The results are displayed in Fig. 8, which demonstrates a clear upward trend in the number of unique tools mentioned per year. Specifically, the number of unique tools referenced in publications has more than doubled since 2011, with a peak of 11 different tools mentioned in 2021. This metric provides valuable insight into the growth and development of the field, and suggests a continued interest and investment in developing new software tools for software visualization.

To complement this analysis, Fig. 9 shows a Gantt chart with the number of publications per tool, only for those tools with more than one publication, and in which years they have been published. *CodeCity* (Wettel and Lanza, 2007b) is the second one with more publications, after *Explorviz* (Fittkau et al., 2015) with 11 publications

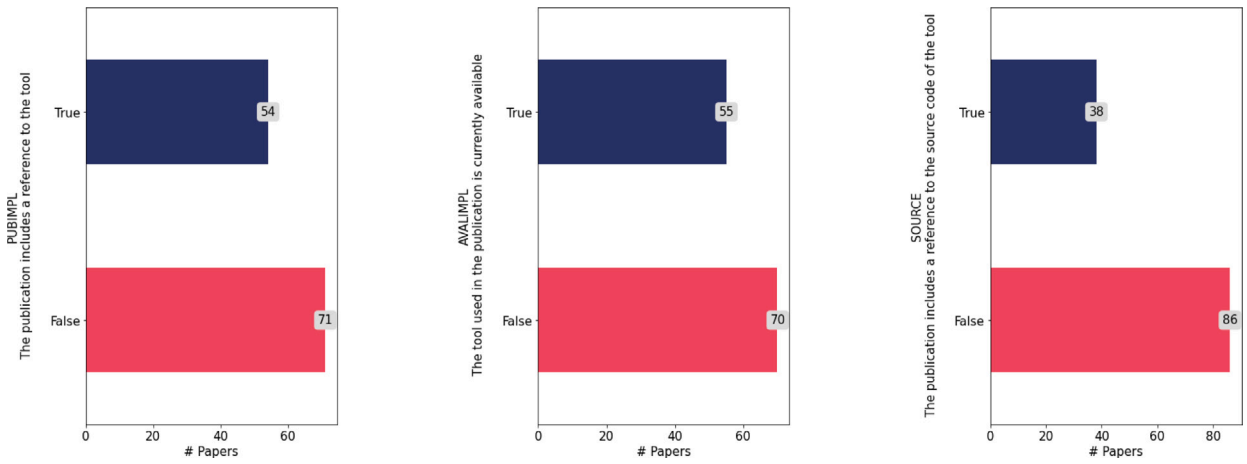

Fig. 10. Publications with references to the tool used, if the tool is available when this analysis was done, and if the publication has a reference to the source code directly.

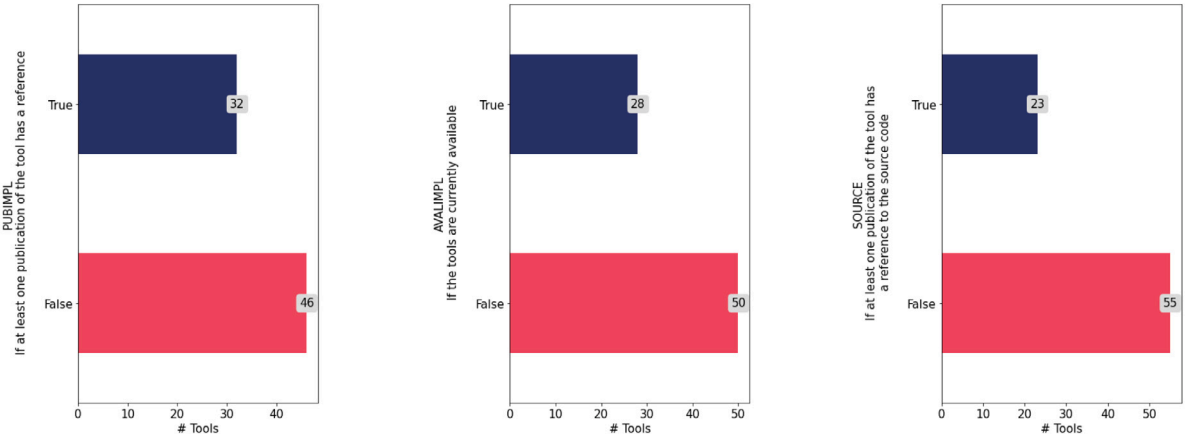

Fig. 11. Tools with at least one reference in all of their publications, if they are available when this analysis was done, and if the tool has a reference to the source code in one of their publication.

in the last 8 years. In this diagram, we see that there are at least 6 tools published in 2022, which is another indication that the research field is very active and the community produces software that currently has more than one publication.

In order to analyze the use of software visualization tools in publications, we examined several metrics, as depicted in Fig. 10. The majority of publications did not have a reference to the code within the publication itself, and many of these references were not accessible at the time of analysis, making the replication of these studies difficult.

When analyzing individual tools, as shown in Fig. 11, we found similar results: more than half of the tools did not have a reference in any of their publications, and only approximately 35% of these tools were currently available for execution. Regarding source code availability, the difference was even greater, with only approximately 25% of tools having clear references to their source code. These results provide insight into the accessibility and reproducibility of research in software visualization, and suggest a need for increased attention to the documentation and sharing of code in the field.

RQ_5: *Are the prototypes exploring new emerging immersive technologies?*

Upon closer inspection of Fig. 13, it is evident that over 80% of the software visualization tools analyzed in this study were designed to work on a conventional 2D screen. The use of Virtual Reality (VR) and Augmented Reality (AR) technologies in software visualization has gained significant interest in recent years. As depicted in Fig. 12, the trend shows an increase in the use of these technologies, which can be attributed to their recent emergence in the field. These technologies offer potential benefits such as enhanced immersion, interactivity, and user engagement. The potential of these technologies in software visualization is enormous, as they can provide new and innovative ways of visualizing complex data, making it easier to understand and comprehend. Moreover, with the recent advancements in technology, VR and AR devices have become more accessible, and their cost has reduced significantly, making them more feasible for general use. The use of web-based VR and AR technologies has also increased, enabling the visualization of data on any device with a modern web browser. Therefore, it is clear that the use of VR and AR technologies in software visualization is a promising avenue for research, and future studies could further explore their potential benefits in this field.

Observing the other characteristics of the figure, over 50% of the tools produce a modular framework (for easy adaptation with other elements), over 55% are configurable (they allow interaction and navigation with the user), and more than half of them allow filtering of the data they show. These characteristics are of great importance for a good interaction with the user, since for a good understanding of the data, the interaction and user experience are of great importance.

The number of available metrics and the number of metrics that can represent the tool simultaneously is of major reference in software visualization. In this case, we can see that in Fig. 13 there is a large percentage of tools that exceed five available metrics. But only a few of them can represent so many at the same time, being three simultaneous metrics the majority.

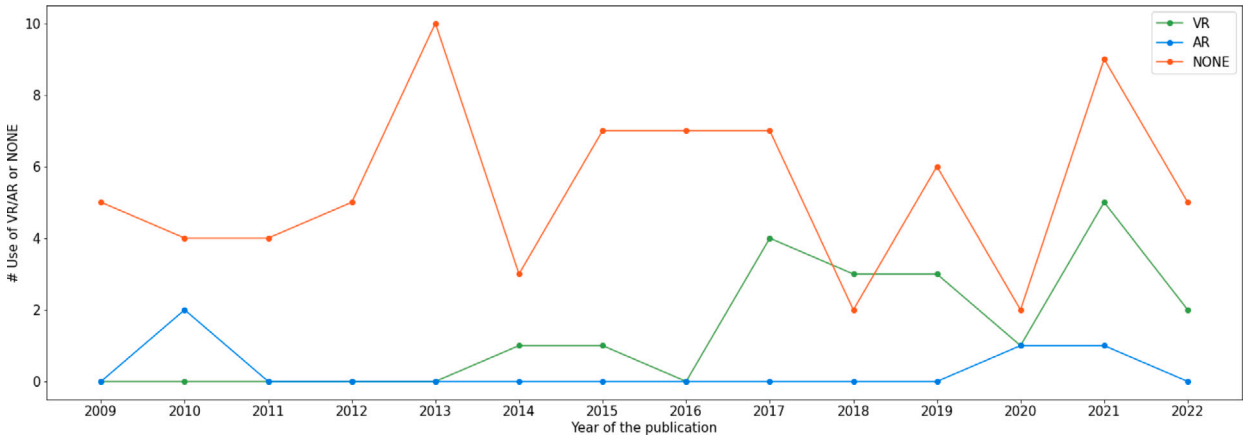

Fig. 12. Number of tools that used VR or AR in years, from 2008 onwards.

6. Discussion

In this paper, we have analyzed and studied the use of the city metaphor and its derivatives in software visualization research. We have shown that this field and the city metaphor are certainly relevant and still growing, not limited to a small audience, and can be found in different publications over the years. In this regard, we can see its study as a case study of how a software engineering practice spreads across academia. The limitations of the metaphor are known and this has led academia to explore other metaphors and improve the tools derived from the studies.

The city metaphor has served as a potent tool in software visualization for numerous years. In this metaphor, software code is metaphorically mapped onto a city, with distinct components represented as buildings and roads. The primary objective is to create a visually intuitive representation of the software, catering even to non-technical individuals. By adopting the city metaphor, complex software structures and architectures can be communicated seamlessly, allowing users to explore the software code as if navigating through a city. Beyond mere visualization, the metaphor empowers software developers to analyze and optimize code structures, pinpointing potential bottlenecks and areas for enhancement. This approach proves highly effective in aiding users in comprehending intricate software systems and assisting developers in crafting more efficient and maintainable code (Yano and Matsuo, 2017). Over time, diverse adaptations and derivatives of the city metaphor have surfaced in software visualization. For example, certain approaches incorporate a landscape metaphor, portraying software components as hills and valleys (Steinbrückner and Lewerentz, 2010b; Kuhn et al., 2012; Steinbrückner, 2013), where elevated peaks signify more critical or complex code. This shift in metaphor enables the integration of additional metrics into the visualization, such as using the peaks and valleys to represent complexity. This feature is not feasible in the traditional city metaphor. Another variation is the solar system metaphor (Hoff et al., 2022), wherein software components are compared to stars and clusters, providing insights into the interconnectedness and relationships among different parts of the software system. The solar system metaphor allows for the representation of multiple systems simultaneously, even incorporating more than one city on every planet, thereby expanding the capacity of the city metaphor and enhancing its scalability. The island metaphor (Misiak et al., 2018) employs a similar concept, enhancing scalability by enabling the representation of cities on different islands. Additionally, certain approaches introduce a game-like interface, enabling users to navigate through software code as if engaged in a video game, exemplified by *CodeMetropolis* (Balogh et al., 2016). These diverse variations and derivatives of the city metaphor present alternative means of visualizing software systems, catering to diverse users and use cases, as demonstrated by the work of Weninger et al. (2019, 2020) in the context of analyzing memory leaks during the evolution of a software city. Despite their differences, these approaches collectively share the overarching goal of making complex software structures more accessible and understandable to a broader audience.

> **RQ_1** summary: The city metaphor remains a vibrant approach for software visualization. This metaphor involves the mapping of software code onto a city, where different components are represented as buildings and roads. The goal is to create an intuitive visual representation that assists developers in optimizing code and aids users in comprehending complex software systems. Additionally, variations and derivatives of the city metaphor offer diverse ways to visualize software systems.

As detailed in Section 5.1, our analysis underscores the continuous growth and dynamism of the software visualization research domain. Each passing year witnesses the emergence of new publications and innovative approaches within this field. This trend is not merely coincidental; rather, it is driven by the escalating complexity and centrality of software systems in our daily lives. As software assumes an ever-increasing role in critical domains, the imperative for effective tools and techniques to comprehend and visualize software code becomes more pronounced. Researchers specializing in software visualization are at the forefront of pioneering novel avenues for achieving this goal. For instance, researchers are actively exploring the integration of cutting-edge technologies, such as augmented reality (AR), into their visualization methods. Hansen demonstrated an augmented reality implementation of *ExplorViz*, utilizing markers with a device equipped with a camera, highlighting the effectiveness of presenting software cities in this immersive environment (Hansen, 2021). Another noteworthy example of augmented reality application in software presentation is the tool *RobotTIC* (Schez-Sobrino et al., 2020), a serious game that combines gamification and AR to facilitate programming learning for students in lower levels of the education system. Moreover, the research community is dedicated to enhancing the symbiosis between software visualization tools and the software development process itself (Alshakhouri, 2013). Such efforts aim to facilitate developers in their endeavors to scrutinize and optimize their code, thereby enhancing software quality and performance (Ogami et al., 2017). The escalating demand for software visualization tools is evident across various industries, including software development (Panas et al., 2003), exemplified

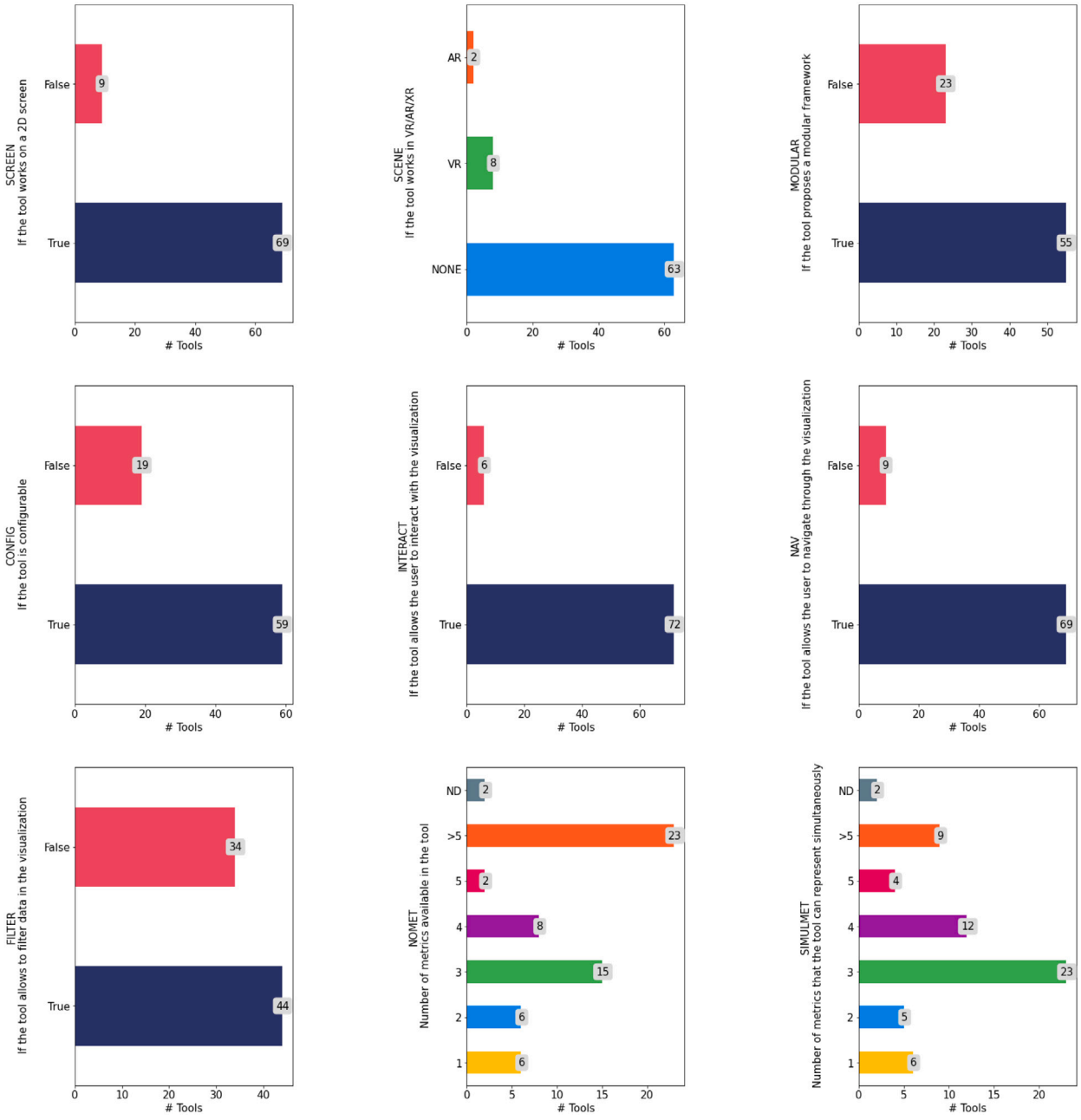

Fig. 13. Different characteristics of the tools. First row: number of tools that work in a 2D conventional screen (left), number of tools that work in Augmented Reality, Virtual Reality or do not work in any of them (middle), and number of tools that propose a modular framework (right). Second row: number of tools that are configurable (left), number of tools that allow interaction (middle), and number of tools that allow navigation inside the visualization (right). Third row: number of tools that allow data filtering (left), number of tools distributed by the number of metrics available to use (middle), and number of tools distributed by the number of metrics that can be represented simultaneously (right).

by the tool *EvoStreets* (Steinbrückner and Lewerentz, 2010b). In the realm of security, the *Secure CodeCity* (Abeysinghe et al., 2021) tool plays a prominent role. Additionally, in the field of data science, the *BabiaXR* (Moreno-Lumbreras et al., 2022) tool stands out as an illustrative example. These instances underscore the growing importance of research in this field. This growth is a testament to the increasing recognition of software visualization's pivotal role in advancing computer science. As more researchers from diverse backgrounds and fields of expertise join the fold of software visualization, the discipline stands poised to perpetuate its expansion and make enduring contributions to the broader landscape of computer science.

RQ_2 summary: Our study reveals varying levels of authorship and collaboration in software visualization research. While some authors contribute sporadically, others form tightly-knit subcommunities. Notably, *CodeCity* authors exhibit significant influence. However, the concentration of active researchers within specific groups underscores the need for greater diversity and inclusivity in this dynamic field.

The city metaphor stands out as the most widely employed metaphor in the field, with 89 studies incorporating its use. This prevalence suggests ongoing research interest, with scholars exploring various aspects and advantages of this visualization method. Researchers have ventured into different algorithms for layout design, as exemplified by the work of *BabiaXR* (Moreno-Lumbreras et al., 2023), where the spiral algorithm is employed to position buildings and quarters. Notably, several other metaphors are referenced in only a single study, hinting at ongoing validation efforts. For instance, Zielasko et al. (2022) introduced the cubes metaphor in 2022, indicating potential ongoing validation work. Conversely, some metaphors referenced in only one study may have been abandoned, perhaps suggesting inadequacy for effectively representing the presented data. An example is the particle metaphor utilized by Scarle and Walkinshaw (2015) in a single study from 2015, which might indicate limited success or applicability.

In delving into the characteristics of the studies, it becomes evident that a considerable number of them introduce novel concepts, marking advancements in the state of the art. This pursuit of innovation is paramount, particularly in a tech-driven landscape where software systems continuously grow in complexity. Researchers keenly acknowledge the imperative to adapt and refine visualization techniques to keep abreast of these advancements. The consistent emergence of various adaptations of the CodeCity tool underscores its enduring significance as a foundational software visualization application. This adaptability is showcased in diverse contexts, such as dependencies analysis (Yano and Matsuo, 2017; Caserta et al., 2011), testing (Hatko et al., 2012; Borg et al., 2018), and concurrency (Waller et al., 2013). Researchers are evidently drawn to this tool, seeking to build upon its strengths and address specific needs within the software visualization community. Additionally, the introduction of new implementations of the city metaphor reflects a commitment to diversity and innovation (Fittkau et al., 2013; Khaloo et al., 2017; Dugerdil and Alam, 2008; Moreno-Lumbreras et al., 2022). Given the substantial variations in software systems' complexity and structure, it is evident that one-size-fits-all visualization techniques fall short. The adaptable nature of the city metaphor continues to evolve through fresh implementations that cater to distinct facets of software analysis. Moreover, the exploration of new use cases signifies the city metaphor's ability to transcend its initial boundaries (Steinbrückner and Lewerentz, 2010c; Schreiber and Brüggemann, 2017; Erra and Scanniello, 2012). Researchers increasingly recognize its applicability across domains beyond its original scope, ranging from system maintenance to code quality assessment. This adaptability positions the city metaphor as a versatile and practical tool, addressing a wide spectrum of software visualization requirements.

RQ_3 summary: The city metaphor, employed in 89 studies, stands as the predominant choice in software visualization, showcasing sustained research interest and exploration of its diverse aspects. While some metaphors show signs of validation efforts or potential abandonment, the city metaphor's enduring popularity reflects its adaptability and continuous evolution. Researchers leverage its versatility, evident in the diverse applications, variations of the CodeCity tool, and exploration of new use cases, positioning it as a dynamic and foundational tool in software visualization.

Moreover, the landscape of software visualization has been significantly shaped by recent technological advancements, fostering a conducive environment for researchers to develop software tools. As elucidated in Section 5.2, this trend has led to a proliferation of software tools within the software visualization field. The increased accessibility of technology has substantially lowered the barriers to entry, enabling researchers to conceive and create new applications and frameworks without necessitating extensive resources, thus rendering the field less resource-intensive than in previous eras. This burgeoning repository of software artifacts, fostered by researchers in the software visualization domain, plays a pivotal role in propelling the state-of-the-art within the discipline. It not only equips software developers with practical tools for enhancing the quality and maintainability of their code but also underscores the imperative of transparent and reproducible research practices. Within the confines of this specialized and highly technical field, the algorithms and techniques employed in research studies often attain a level of complexity that demands access to the original software code for faithful reproduction. By proactively making their software resources readily available and transparently referenced in their research publications, researchers not only facilitate the understanding and extension of their work by peers but also ensure due recognition for their contributions. Furthermore, this practice promotes a culture of validation and rigorous testing by fellow researchers, ultimately enhancing the overall quality and reliability of research endeavors in the field.

RQ_4 summary: Technological progress has broadened the avenues for researchers to create software tools, thereby fostering the proliferation of tools within the software visualization domain. This expansion not only propels the state-of-the-art but also equips software developers with tangible tools to enhance the quality and manageability of their code.

Our analysis of software visualization tools has unveiled the nascent but promising emergence of virtual and augmented reality (VR/AR) environments as a novel approach within the field. While VR/AR promises substantial advantages, including heightened immersion and improved perceptual capabilities, it is not without its attendant challenges, such as the requirement for specialized hardware and software infrastructure and the development of effective 3D interaction techniques. However, researchers are engaged in addressing these hurdles and innovating new applications for VR/AR in software visualization. With the relentless pace of technological advancement and the growing accessibility of immersive technologies like extended reality (XR), we anticipate a burgeoning integration of these technologies into software visualization tools (Hansen, 2021; Schez-Sobrino et al., 2020; Moreno-Lumbreras et al., 2021; Hoff et al., 2022; Aoki et al., 2021; Misiak et al., 2018; Merino et al., 2017). This evolution has the potential to revolutionize the paradigms through which we analyze and comprehend software code. Consequently, the current trajectory of incorporating XR technology in software visualization appears poised for sustained growth, with the prospect of a proliferation of tools that harness these immersive environments to enhance our understanding and interaction with software systems. This trend not only represents a significant advancement within the field but also underscores the inherent value of our investigation into the evolving landscape of software visualization.

RQ_5 summary: The use of virtual and augmented reality in software visualization is a relatively new approach, and while it presents challenges, such as the need for specialized hardware and software, researchers are actively exploring these challenges and devising new ways to utilize these technologies for software visualization, with the trend expected to continue growing as new immersion technologies, such as extended reality, become more accessible.

7. Threats to validity

Following the main types of validity threats in Experimentation Software Engineering (ESE) proposed by Wohlin et al. (2012), we discuss the four types: conclusion, internal, construct, and external threats.

In this case, as it is an SMS, the validity of the conclusion, which is related to how sure we are about the treatment we use in an investigation does not affect this study.

Internal validity is the extent to which a causal conclusion based on a study is warranted, which is determined by the degree to which a study minimizes systematic errors. We have attempted to minimize this threat by following the guidelines of Kitchenham et al. (2002) and Petersen et al. (2008), and offering a replication package so that everybody can replicate the study, available in Section 8. However, there might be a selection bias due to having chosen Google Scholar as the source of all publications. We have tried to validate this threat by conducting a study of the quality of the chosen sample, taking references from various related articles, and verifying that they were in our sample. We also checked that the results in other databases were included in the Google Scholar sample. Another internal threat may be the choice to search for studies that cite two articles; there may be more outside this scope. We think this is a minor threat since the *CodeCity* (Wettel and Lanza, 2007b) tool is the most used in the field and the most cited, even one of the authors being the largest publisher in this field, if this tool is not referenced in a study related to the visualization of software and the metaphor of the city we can assume that this paper is out of scope. The same is true for the first article that studies the metaphor of the city in software visualization, *Software World* (Knight and Munro, 1999), being the basis of any study and referenced in subsequent studies, every publication should reference the first study in the literature about the field.

Construct validity is the degree to which an investigation measures what it claims to be measuring. In this case, we measure the activity of the research field in number of publications and tools derived from the publications over time, in order to be able to observe if the field is growing or decreasing and if it is still active. There may be human errors in feature extraction, but we have tried to mitigate this by first performing a multi-author category analysis with two authors extracting all features from the included studies and a one checking the difference between these two. Our replication package does not eliminate human errors that we may have incurred, but it offers the possibility for others to check and improve our work.

External validity is the degree to which results can be generalized to other contexts. We cannot claim that our results can be generalized to Empirical Software Engineering research, since we have selected a specific case of use of the city metaphor, in this case in software visualization. However, the value of case studies should be not undermined.

8. Conclusions

We reviewed a set of 404 paper that referenced *Software World* (Knight and Munro, 1999) and *CodeCity* (Wettel and Lanza, 2007b). We extracted evaluation strategies, data collection methods, and other various aspects of evaluations, obtaining a final sample of 168 studies. We found that 120 papers present an advance in the state of the art in the field of research, which represents more than 75% of the papers included. 91 papers present a variation of the city metaphor using the most cited tool/study, *CodeCity*. 118 of these papers present a new implementation or improvement of a previously presented tool, and 119 present a new use case of the visualization of the city metaphor or its derivatives. We have presented an analysis of the evolution of the research field, demonstrating the number of publications per year. The number of publications has increased over time, with a clear turning point in 2007. Specifically, we observed that the number of publications in 2022 was twice as high as the number of publications in 2007, indicating a growing interest in this research field. The presented data highlights the active and stable nature of this field within the research community. This has also had an impact on the growth of the research community, increasing considerably, and maintaining groups that continue to publish in the same field.

We also analyzed and studied the tools derivated from the studies, having a total of 78 tools. We proved that researchers are producing more software, being the peak in 2021 with 11 tools. The city metaphor is still one of the most used metaphors in software visualization. However, only 17 tools appear in more than one publication, which may indicate a certain abandonment of these tools. A more specific study will be needed to know the reason. Regarding the availability of the tools, only 35% of these tools are currently available for execution. Regarding the reference to the source code in the publication, the difference is even greater, and only approximately 25% have clear references to the source code of the tool. Finally, only a few tools are exploring other environments apart from the 2D screens, such as extended reality.

Summarizing, we observed that the growth of the research community in this field has been facilitated by the increasing accessibility of new technology. This has enabled researchers to explore new approaches and techniques in software visualization, which were previously considered too resource-intensive. We found evidence of this trend in the growing number of publications and tools, which suggests that the community is becoming more active and diverse. We believe that this trend will continue as new technologies emerge and the field of software visualization evolves. Ultimately, we hope that our study will contribute to the development of new approaches and tools in software visualization, and that it will inspire researchers to continue exploring this exciting and rapidly evolving field.

We believe that our study will help (a) researchers to reflect on the design of new approaches for using new metaphors in software visualization, and (b) developers to be aware of the existing tools and their features, explaining the characteristics for the benefit of the proposed software visualization approaches.

Replication package

The data and the analysis obtained for our study, including the material needed for replication, are publicly available.[5]

CRediT authorship contribution statement

David Moreno-Lumbreras: Data curation, Formal analysis, Funding acquisition, Investigation, Project administration, Visualization, Writing – original draft, Writing – review & editing, Conceptualization, Methodology. **Jesus M. Gonzalez-Barahona:** Funding acquisition, Investigation, Methodology, Project administration, Supervision, Validation. **Gregorio Robles:** Data curation, Funding acquisition, Investigation, Project administration, Supervision, Validation, Visualization, Writing – review & editing. **Valerio Cosentino:** Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Validation.

[5] Replication Package: https://doi.org/10.5281/zenodo.10492845.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: David Moreno-Lumbreras reports financial support was provided by Community of Madrid. Jesus M. Gonzalez-Barahona reports financial support was provided by Community of Madrid. Gregorio Robles reports financial support was provided by Madrid Regional Government. Gregorio Robles reports financial support was provided by EU Structural Funds.

Data availability

Data is available in a replication package hosted in Zenodo, linked in the paper.

Acknowledgments

We acknowledge the financial support of the Community of Madrid, Spain for the project IND2018/TIC-9669, and the Spanish Government for the project RTI-2018-101963-B-100, and of the Madrid Regional Government, Spain (e-Madrid-CM - P2018/TCS-4307), co-financed by EU Structural Funds (FSE and FEDER).

References

Abeysinghe, AA.T.G., Shalika, M.A.S., Ahamed, M.S.N., Mufarrij, S.M., 2021. Secure CodeCity a Framework for Security Vulnerability Visualization (Ph.D. thesis).

Alshakhouri, Mujtaba Alawi J., 2013. ScrumCity: Synchronised Visualisation of Software Process and Product Artefacts (Ph.D. thesis). Auckland University of Technology.

Aoki, Tatsushi, Tsuji, Airi, Fujinami, Kaori, 2021. Software structure exploration in an immersive virtual environment. In: 2021 IEEE 10th Global Conference on Consumer Electronics. GCCE, IEEE, pp. 946–947.

Averbukh, Vladimir, 2001. Visualization metaphors. Program. Comput. Softw. 27, 227–237.

Balogh, Gergo, Gergely, Tamas, Beszedes, Arpad, Gyimothy, Tibor, 2016. Using the city metaphor for visualizing test-related metrics. In: 2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering. SANER, Vol. 2, pp. 17–20.

Balzer, Michael, Noack, Andreas, Deussen, Oliver, Lewerentz, Claus, 2004. Software landscapes: Visualizing the structure of large software systems. In: Deussen, Oliver, Hansen, Charles, Keim, Daniel, Saupe, Dietmar (Eds.), Eurographics / IEEE VGTC Symposium on Visualization. The Eurographics Association.

Basit, Hamid Abdul, Hammad, Muhammad, Koschke, Rainer, 2015. A survey on goal-oriented visualization of clone data. In: 2015 IEEE 3rd Working Conference on Software Visualization. VISSOFT, IEEE, pp. 46–55.

Bassil, Sarita, Keller, Rudolf K., 2001. Software visualization tools: Survey and analysis. In: Proceedings 9th International Workshop on Program Comprehension. IWPC 2001, IEEE, pp. 7–17.

Baum, D., Schilbach, J., Kovacs, P., Eisenecker, U., Müller, R., 2017. GETAVIZ: Generating structural, behavioral, and evolutionary views of software systems for empirical evaluation. In: 2017 IEEE Working Conference on Software Visualization. VISSOFT, pp. 114–118.

Borg, Markus, Brytting, Daniel, Hansson, Daniel, 2018. Enabling visual design verification analytics-from prototype visualizations to an analytics tool using the unity game engine. arXiv preprint arXiv:1808.09767.

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

Capece, Nicola, Erra, Ugo, Romano, Simone, Scanniello, Giuseppe, 2017. Visualising a software system as a city through virtual reality. In: De Paolis, Lucio Tommaso, Bourdot, Patrick, Mongelli, Antonio (Eds.), Augmented Reality, Virtual Reality, and Computer Graphics. Springer International Publishing, Cham, pp. 319–327.

Carpendale, Sheelagh, Ghanam, Yaser, 2008. A Survey Paper on Software Architecture Visualization. Technical Report, University of Calgary.

Caserta, Pierre, Zendra, Olivier, 2010. Visualization of the static aspects of software: A survey. IEEE Trans. Vis. Comput. Graph. 17 (7), 913–933.

Caserta, Pierre, Zendra, Olivier, 2011. Visualization of the static aspects of software: A survey. IEEE Trans. Vis. Comput. Graphics 17 (7), 913–933.

Caserta, Pierre, Zendra, Olivier, Bodénes, Damien, 2011. 3D hierarchical edge bundles to visualize relations in a software city metaphor. In: 2011 6th International Workshop on Visualizing Software for Understanding and Analysis. VISSOFT, IEEE, pp. 1–8.

Diehl, Stephan, 2007. Software Visualization: Visualizing the Structure, Behaviour, and Evolution of Software. Springer Science & Business Media.

Donalek, Ciro, Djorgovski, S. G., Cioc, Alex, Wang, Anwell, Zhang, Jerry, Lawler, Elizabeth, Yeh, Stacy, Mahabal, Ashish, Graham, Matthew, Drake, Andrew, Davidoff, Scott, Norris, Jeffrey S., Longo, Giuseppe, 2014. Immersive and collaborative data visualization using virtual reality platforms. In: 2014 IEEE International Conference on Big Data (Big Data). pp. 609–614.

Dugerdil, Philippe, Alam, Sazzadul, 2008. Execution trace visualization in a 3D space. In: Fifth International Conference on Information Technology: New Generations (Itng 2008). IEEE, pp. 38–43.

Erra, Ugo, Scanniello, Giuseppe, 2012. Towards the visualization of software systems as 3D forests: The CodeTrees environment. In: Proceedings of the 27th Annual ACM Symposium on Applied Computing. SAC '12, Association for Computing Machinery, New York, NY, USA, pp. 981–988.

Fittkau, Florian, Krause, Alexander, Hasselbring, Wilhelm, 2015. Exploring software cities in virtual reality. In: 2015 IEEE 3rd Working Conference on Software Visualization. VISSOFT, pp. 130–134.

Fittkau, Florian, Waller, Jan, Wulf, Christian, Hasselbring, Wilhelm, 2013. Live trace visualization for comprehending large software landscapes: The ExplorViz approach. In: 2013 First IEEE Working Conference on Software Visualization. VISSOFT, IEEE, pp. 1–4.

Graham, Hamish, Yang, Hong Yul, Berrigan, Rebecca, 2004. A solar system metaphor for 3D visualisation of object oriented software metrics. In: Churcher, Neville, Churcher, Clare (Eds.), Australasian Symposium on Information Visualisation, InVis.Au, Christchurch, New Zealand, 23-24 January 2004. In: CRPIT, vol. 35, Australian Computer Society, pp. 53–59.

Hamou-Lhadj, Abdelwahab, Lethbridge, Timothy C., 2004. A survey of trace exploration tools and techniques. In: Proceedings of the 2004 Conference of the Centre for Advanced Studies on Collaborative Research. pp. 42–55.

Hansen, Malte, 2021. Collaborative Program Comprehension Based on Augmented Reality (Ph.D. thesis). Kiel University.

Hatko, Reinhard, Baumeister, Joachim, Puppe, Frank, 2012. CoverageCity: Test coverage for clinical guidelines. In: KESE@ ECAI. Citeseer.

Hoff, Adrian, Gerling, Lea, Seidl, Christoph, 2022. Utilizing software architecture recovery to explore large-scale software systems in virtual reality. In: 2022 Working Conference on Software Visualization. VISSOFT, pp. 119–130.

Isaacs, Katherine E, Giménez, Alfredo, Jusufi, Ilir, Gamblin, Todd, Bhatele, Abhinav, Schulz, Martin, Hamann, Bernd, Bremer, Peer-Timo, 2014. State of the art of performance visualization. EuroVis (STARs).

Khaloo, Pooya, Maghoumi, Mehran, Taranta, Eugene, Bettner, David, Laviola, Joseph, 2017. Code park: A new 3d code visualization tool. In: 2017 IEEE Working Conference on Software Visualization. VISSOFT, IEEE, pp. 43–53.

Kitchenham, B.A., Pfleeger, S.L., Pickard, L.M., Jones, P.W., Hoaglin, D.C., El Emam, K., Rosenberg, J., 2002. Preliminary guidelines for empirical research in software engineering. IEEE Trans. Softw. Eng. 28 (8), 721–734.

Knight, C., Munro, M., 1999. Comprehension with[in] virtual environment visualisations. In: Proceedings Seventh International Workshop on Program Comprehension. pp. 4–11.

Koschke, Rainer, 2003. Software visualization in software maintenance, reverse engineering, and re-engineering: a research survey. J. Softw. Maint. Evol.: Res. Pract. 15 (2), 87–109.

Kuhn, Adrian, Loretan, Peter, Nierstrasz, Oscar, 2012. Consistent layout for thematic software maps. CoRR abs/1209.5490.

Lakoff, George, Johnson, Mark, 1980. Metaphors We Live By. University of Chicago Press.

Langelier, Guillaume, Sahraoui, Houari, Poulin, Pierre, 2005. Visualization-based analysis of quality for large-scale software systems. In: Proceedings of the 20th IEEE/ACM International Conference on Automated Software Engineering. ASE '05, Association for Computing Machinery, New York, NY, USA, pp. 214–223.

Löwe, Welf, Ericsson, Morgan, Lundberg, Jonas, Panas, Thomas, Petersson, Niklas, 2003. VizzAnalyzer- A software comprehension framework.

Maletic, J.I., Leigh, J., Marcus, A., Dunlap, G., 2001. Visualizing object-oriented software in virtual reality. In: Proceedings 9th International Workshop on Program Comprehension. IWPC 2001, pp. 26–35.

Marcus, Andrian, Feng, Louis, Maletic, Jonathan I., 2003. 3D representations for software visualization. In: Proceedings of the 2003 ACM Symposium on Software Visualization. SoftVis '03, Association for Computing Machinery, New York, NY, USA, pp. 27–ff.

Maruyama, Katsuhisa, Omori, Takayuki, Hayashi, Shinpei, 2014. A visualization tool recording historical data of program comprehension tasks. In: Proceedings of the 22nd International Conference on Program Comprehension. In: ICPC 2014, Association for Computing Machinery, New York, NY, USA, pp. 207–211.

Mattila, Anna-Liisa, Ihantola, Petri, Kilamo, Terhi, Luoto, Antti, Nurminen, Mikko, Väätäjä, Heli, 2016. Software visualization today: Systematic literature review. In: Proceedings of the 20th International Academic Mindtrek Conference. AcademicMindtrek '16, Association for Computing Machinery, New York, NY, USA, pp. 262–271.

Merino, Leonel, Fuchs, Johannes, Blumenschein, Michael, Anslow, Craig, Ghafari, Mohammad, Nierstrasz, Oscar, Behrisch, Michael, Keim, Daniel A., 2017. On the impact of the medium in the effectiveness of 3D software visualizations. In: 2017 IEEE Working Conference on Software Visualization. VISSOFT, pp. 11–21.

Merino, L., Ghafari, M., Anslow, C., Nierstrasz, O., 2017. CityVR: Gameful software visualization. In: 2017 IEEE International Conference on Software Maintenance and Evolution. ICSME, pp. 633–637.

Merino, Leonel, Ghafari, Mohammad, Anslow, Craig, Nierstrasz, O., 2018. A systematic literature review of software visualization evaluation. J. Syst. Softw. 144.

Misiak, Martin, Schreiber, Andreas, Fuhrmann, Arnulph, Zur, Sascha, Seider, Doreen, Nafeie, Lisa, 2018. IslandViz: A tool for visualizing modular software systems in virtual reality. In: 2018 IEEE Working Conference on Software Visualization. VISSOFT, pp. 112–116.

Moreno-Lumbreras, David, Gonzalez-Barahona, Jesus M, Villaverde, Andrea, 2022. BabiaXR: Virtual reality software data visualizations for the Web. In: 2022 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops. VRW, IEEE, pp. 71–74.

Moreno-Lumbreras, David, Minelli, Roberto, Villaverde, Andrea, Gonzalez-Barahona, Jesus M., Lanza, Michele, 2023. CodeCity: A comparison of on-screen and virtual reality. Inf. Softw. Technol. 153, 107064.

Moreno-Lumbreras, David, Robles, Gregorio, Izquierdo-Cortázar, Daniel, Gonzalez-Barahona, Jesus M, 2021. To VR or not to VR: Is virtual reality suitable to understand software development metrics? arXiv preprint arXiv:2109.13768.

Novais, Renato Lima, Torres, André, Mendes, Thiago Souto, Mendonça, Manoel, Zazworka, Nico, 2013. Software evolution visualization: A systematic mapping study. Inf. Softw. Technol. 55 (11), 1860–1883.

O'Donovan, Peter, Leahy, Kevin, Bruton, Ken, O'Sullivan, Dominic T.J., 2015. Big data in manufacturing: a systematic mapping study. J. Big Data 2 (1), 20.

Ogami, Katsuya, Kula, Raula Gaikovina, Hata, Hideaki, Ishio, Takashi, Matsumoto, Kenichi, 2017. Using high-rising cities to visualize performance in real-time. In: 2017 IEEE Working Conference on Software Visualization. VISSOFT, IEEE, pp. 33–42.

Panas, T., Berrigan, R., Grundy, J., 2003. A 3D metaphor for software production visualization. In: Proceedings on Seventh International Conference on Information Visualization, 2003. IV 2003, pp. 314–319.

Panas, Thomas, Epperly, Thomas, Quinlan, Daniel, Saebjornsen, Andreas, Vuduc, Richard, 2007. Communicating software architecture using a unified single-view visualization. In: 12th IEEE International Conference on Engineering Complex Computer Systems (ICECCS 2007). pp. 217–228.

Panas, Thomas, Lincke, Rüdiger, Löwe, Welf, 2005. Online-configuration of software visualizations with Vizz3D. In: Proceedings of the 2005 ACM Symposium on Software Visualization. SoftVis '05, Association for Computing Machinery, New York, NY, USA, pp. 173–182.

Paredes, Julia, Anslow, Craig, Maurer, Frank, 2014. Information visualization for agile software development. In: 2014 Second Ieee Working Conference on Software Visualization. IEEE, pp. 157–166.

Petersen, Kai, Feldt, Robert, Mujtaba, Shahid, Mattsson, Michael, 2008. Systematic mapping studies in software engineering. In: Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering. EASE '08, BCS Learning Development Ltd., Swindon, GBR, pp. 68–77.

Pfahler, Federico, Minelli, Roberto, Nagy, Csaba, Lanza, Michele, 2020. Visualizing evolving software cities. In: 2020 Working Conference on Software Visualization. VISSOFT, pp. 22–26.

Rodrigues, Luís Filipe, Oliveira, Abílio, Rodrigues, Helena, 2019. Main gamification concepts: A systematic mapping study. Heliyon 5 (7), e01993.

Scarle, Simon, Walkinshaw, Neil, 2015. Visualising software as a particle system. In: 2015 IEEE 3rd Working Conference on Software Visualization. VISSOFT, pp. 66–75.

Scarsbrook, J. D., Ko, R. K. L., Rogers, B., Bainbridge, D., 2018. MetropolJS: Visualizing and debugging large-scale JavaScript program structure with treemaps. In: 2018 IEEE/ACM 26th International Conference on Program Comprehension. ICPC, pp. 389–3893.

Scheibel, Willy, Limberger, Daniel, Döllner, Jürgen, 2020. Survey of treemap layout algorithms. http://dx.doi.org/10.1145/3430036.3430041.

Scheibel, Willy, Weyand, Christopher, Döllner, Jürgen, 2018. EvoCells: A treemap layout algorithm for evolving tree data. In: VISIGRAPP.

Schez-Sobrino, Santiago, Vallejo, David, Glez-Morcillo, Carlos, Redondo, Miguel Á, Castro-Schez, José Jesús, 2020. RoboTIC: A serious game based on augmented reality for learning programming. Multimedia Tools Appl. 79, 34079–34099.

Schots, Marcelo, Vasconcelos, Renan, Werner, Cláudia, 2014. A Quasi-Systematic Review on Software Visualization Approaches for Software Reuse. Technical Report.

Schreiber, Andreas, Brüggemann, Marlene, 2017. Interactive visualization of software components with virtual reality headsets. In: 2017 IEEE Working Conference on Software Visualization. VISSOFT, pp. 119–123.

Seriai, Abderrahmane, Benomar, Omar, Cerat, Benjamin, Sahraoui, Houari, 2014. Validation of software visualization tools: A systematic mapping study. In: 2014 Second IEEE Working Conference on Software Visualization. pp. 60–69.

Steinbrückner, Frank, 2013. Consistent software cities: supporting comprehension of evolving software systems.

Steinbrückner, Frank, Lewerentz, Claus, 2010a. Representing development history in software cities. In: Proceedings of the 5th International Symposium on Software Visualization. SOFTVIS '10, Association for Computing Machinery, New York, NY, USA, pp. 193–202.

Steinbrückner, Frank, Lewerentz, Claus, 2010b. Representing development history in software cities. In: Proceedings of the 5th International Symposium on Software Visualization. SOFTVIS '10, Association for Computing Machinery, New York, NY, USA, pp. 193–202.

Steinbrückner, Frank, Lewerentz, Claus, 2010c. Representing development history in software cities. In: Proceedings of the 5th International Symposium on Software Visualization. pp. 193–202.

Storey, Margaret-Anne D., Čubranić, Davor, German, Daniel M, 2005. On the use of visualization to support awareness of human activities in software development: a survey and a framework. In: Proceedings of the 2005 ACM Symposium on Software Visualization. pp. 193–202.

Teyseyre, Alfredo R., Campo, Marcelo R., 2008. An overview of 3D software visualization. IEEE Trans. Vis. Comput. Graph. 15 (1), 87–105.

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

Vincur, Juraj, Polasek, Ivan, Navrat, Pavol, 2017b. Searching and exploring software repositories in virtual reality. In: Proceedings of the 23rd ACM Symposium on Virtual Reality Software and Technology. VRST '17, Association for Computing Machinery, New York, NY, USA.

Waller, Jan, Wulf, Christian, Fittkau, Florian, Döhring, Philipp, Hasselbring, Wilhelm, 2013. Synchrovis: 3d visualization of monitoring traces in the city metaphor for analyzing concurrency. In: 2013 First IEEE Working Conference on Software Visualization. VISSOFT, IEEE, pp. 1–4.

Weninger, Markus, Makor, Lukas, Mössenböck, Hanspeter, 2019. Memory leak visualization using evolving software cities. Softwaretechnik-Trends 39, 44–46.

Weninger, Markus, Makor, Lukas, Mössenböck, Hanspeter, 2020. Memory cities: Visualizing heap memory evolution using the software city metaphor. In: 2020 Working Conference on Software Visualization. VISSOFT, pp. 110–121.

Wettel, Richard, Lanza, Michele, 2007a. Program comprehension through software habitability. In: 15th IEEE International Conference on Program Comprehension. (ICPC '07), pp. 231–240.

Wettel, Richard, Lanza, Michele, 2007b. Visualizing software systems as cities. In: 2007 4th IEEE International Workshop on Visualizing Software for Understanding and Analysis. pp. 92–99.

Wettel, R., Lanza, M., 2008. Visual exploration of large-scale system evolution. In: 2008 15th Working Conference on Reverse Engineering. pp. 219–228.

Wettel, Richard, Lanza, Michele, 2008. Visually localizing design problems with disharmony maps. In: Proceedings of the 4th ACM Symposium on Software Visualization. SoftVis '08, Association for Computing Machinery, New York, NY, USA, pp. 155–164.

Wohlin, Claes, Runeson, Per, Höst, Martin, Ohlsson, Magnus C, Regnell, Björn, Wesslén, Anders, 2012. Experimentation in Software Engineering. Springer Science & Business Media.

Wolny, Sabine, Mazak, Alexandra, Carpella, Christine, Geist, Verena, Wimmer, Manuel, 2020. Thirteen years of sysml: a systematic mapping study. Softw. Syst. Model. 19 (1), 111–169.

Yano, Keisuke, Matsuo, Akihiko, 2017. Data access visualization for legacy application maintenance. In: 2017 IEEE 24th International Conference on Software Analysis, Evolution and Reengineering. SANER, pp. 546–550.

Young, P., Munro, M., 1998. Visualising software in virtual reality. In: Proceedings. 6th International Workshop on Program Comprehension. IWPC'98 (Cat. No.98TB100242). pp. 19–26.

Zielasko, Jan, Tempel, Soren, Herdt, Vladimir, Drechsler, Rolf, 2022. 3D visualization of symbolic execution traces. In: 2022 Forum on Specification & Design Languages. FDL, pp. 1–8.

David Moreno-Lumbreras: I hold a Ph.D. in Information Technologies and Systems from Universidad Rey Juan Carlos Prior to that, I obtained a Bachelor's degree in Telecommunications Engineering and a Master's degree in Telecommunications Engineering. I specialize in web development, with a strong focus on Extended Reality (XR) and data/software visualization. Prior to this, I gained experience in the industry as a front-end developer and also in backend development using Python, particularly with Django. Additionally, I am passionate about sharing knowledge and actively contribute to open-source software projects.

Jesús M. González-Barahona: I am a Full Professor in Telematics Engineering at Universidad Rey Juan Carlos (Móstoles, Madrid). I earned my degree in Telecommunications Engineering in 1991 and completed a Ph.D. in Telecommunications Engineering in 1998 at Universidad Politécnica de Madrid. With over 20 years of teaching experience, I specialize in computer networks, data transmission, and telematic services and protocols. My research interests revolve around the empirical study of software development, quantitative methods for activity and process analysis, and data visualization in extended reality (VR and AR). Throughout my career, I have led more than 35 international research projects and authored numerous articles and conference papers.

Gregorio Robles: I am a Full Professor at Universidad Rey Juan Carlos in Madrid, with a Ph.D. in Empirical Software Engineering Research on Free/Open Source Software. I have extensive international experience, including post-doc visits to the University of Lincoln (UK) and the University of Trier (Germany). Appointed as Visiting Research Professor at the University of Skövde (Sweden), I have also participated in Erasmus+ exchanges with Chalmers University (Sweden) and the University of Jyväskylä (Finland). In 2012, my research group founded Bitergia, a software analytics spin-off serving industry leaders like the Linux Foundation, Red Hat, and Mozilla.

Valerio Cosentino: I am a software engineer with 10+ years' experience in academia and industry, adept in Python (6+ years), SQL, and NoSQL databases. My expertise spans API development, source code analysis, prototyping, and practical knowledge of distributed systems and ETL processes. Skills include ownership, accountability, attention to detail, collaboration, and transparency. A proficient communicator, I thrive on interacting with people.