2014 Second IEEE Working Conference on Software Visualization

Validation of Software Visualization Tools: A Systematic Mapping Study

Abderrahmane Seriai, Omar Benomar, Benjamin Cerat, Houari Sahraoui
Département IRO
Université de Montréal, Montreal, Canada
{seriaiab,benomaro,ceratben,sahraouh}@iro.umontreal.ca

***Abstract*—Software visualization as a research field focuses on the visualization of the structure, behavior, and evolution of software. It studies techniques and methods for graphically representing these different aspects of software. Interest in software visualization has grown in recent years, producing rapid advances in the diversity of research and in the scope of proposed techniques, and aiding the application experts who use these techniques to advance their own research.**

Despite the importance of evaluating software visualization research, there is little work studying validation methods. As a consequence, it is usually difficult to produce compelling evidence about the effectiveness of software visualization contributions. The goal of this paper is to study the validation techniques performed in the software visualization literature.

We conducted a systematic mapping study of validation methods in software visualization. We consider 752 articles from multiple sources, published between 2000 and 2012, and study the validation techniques of the software visualization articles.

The main outcome of this study is the lack in rigor when validating software visualization tools and techniques. Although software visualization has grown in interest in the last decade, it still lacks the necessary maturity to be properly and thoroughly evaluating its claims. Most article evaluations studied in this paper are qualitative case studies, including discussions about the benefits of the proposed visualizations.

The results help us understand the needs in software visualization validation techniques. They identify the type of evaluations that should be performed to address this deficiency. The specific analysis of SOFTVIS series articles shows that the specialized conference suffers from the same shortage.

I. Introduction

Software structure refers to the organization of system modules and their relations. Software behavior deals with the execution of the program and its dynamic states. Evolution of software considers the development process of the software system [1]. Software visualization is a semi-automatic approach used to understand one of these three aspects of software, with the contribution of developers, by alleviating their cognitive load.

A number of research studies have been conducted in software visualization integrating different methods of evaluation; some are rigorous, and others are less stringent. However, there has been little effort to systematically survey the evaluation methods in software visualization reported in the literature in order to know what has been done and what needs to be done in the field of software visualization, and to understand the evolution of validation techniques in this field.

In this paper, we report on a systematic mapping study that we conducted on validation methods in software visualization in order to classify the state of the art in this research field. Petersen et al. [2] defines a systematic map as a *"method to build a classification scheme and structure a software engineering field of interest. The analysis of results focuses on frequencies of publications for categories within the scheme. Thereby, the coverage of the research filed can be determined. Different facets of the scheme can also be combined to answer more specific research questions."*

The systematic literature review is another meta study that aims to analyze publications in a given research field. It is increasingly used in Software engineering field [3], [4], [5], and differs from a systematic mapping study, mainly in the research goals and level of analysis. Indeed, a systematic review considers the analysis of the results of reviewed papers to establish the state of evidence. However, a systematic mapping focuses principally on classification, conducting thematic analysis and identifying publication areas. In other words, a systematic mapping, such as [6], may address a larger field because it does not require evaluating papers in such detail, while the systematic review does.

We chose to conduct a systematic mapping of software visualization validation publications, to identify the trends in this field. In particular, we aim to determine if the field of interest follows the general software engineering validation trend.

The remaining part of this paper is organized as follows: Section II describes the systematic mapping process. Section III reports the results of the selection step of our study, and section IV presents the results of the mapping study. A specific study for SOFTVIS conference, and its results are presented in section V. Section VI discusses the study limitations. Before concluding with a summary of the main recommendations for future research on software visualization, we present the related work in Section VII.

II. Systematic Mapping Process

In this section, we discuss the main process steps of our systematic mapping study, as defined by Petersen et al. [2]. These include the definition of research questions, conducting the search for relevant papers, screening of papers, keywording of abstracts and data extraction and mapping (see Figure 1).

978-0-7695-5305-4/14 $31.00 © 2014 IEEE
DOI 10.1109/VISSOFT.2014.19

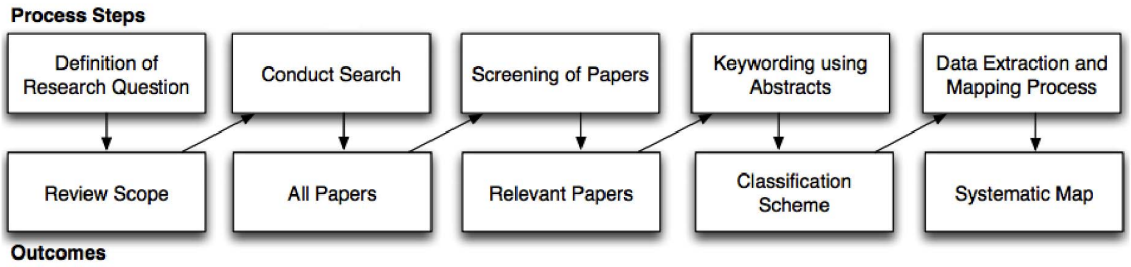

Fig. 1: The systematic mapping process [2].

A. Definition of Research Questions

We reviewed articles on software visualization that propose an evaluation of their approaches to identify major trends in software visualization validation techniques. The main goal of this paper is to determine the quantity and type of techniques used in software visualization validation. To this end, we formulated these research questions:

- **RQ1:** How mature is software visualization research with respect to validation?
- **RQ2:** What validation techniques have been used for the assessment of visualization research?

B. Search Strategy and Data Sources

Starting from the research questions, we extracted keywords to query the primary study digital libraries. The keywords are intended to cover the largest proportion of published material. The main keywords that were initially identified are obviously ”software visualization” and “evaluation”. Moreover, we added the various forms of empirical studies: “experiment”, “case study”, “survey”. Finally, variations and synonyms for these keywords were also considered. To build the search queries, we tried different combinations of the keywords, variations and synonyms with logical operators. An example of such a query is “((case stud* OR valid* OR experi*) AND (software visualization)) “.

We decided to query two publication databases namely: IEEE Xplore(ieeexplre.ieee.org), and Scirus (www.scirus.com) and cover the period spaning from 2000 to 2012. We selected these libraries as they allow us to export the query results with the selected metadata. Other libraries such as Google Scholar and ACM digital library have not been considered because of the absence of this feature. Before starting the review, we also integrated all the papers belonging to the ACM Symposium on Software Visualization (2003-2005-2006-2008-2010) as we noticed that both databases do not index this forum.

We limited the search space to abstract, title, and keywords fields. For some queries, such those involving keywords about study types, the full-text search had to be done. We used such a query in IEEE Xplore, but the Scirus query API didn’t allow for searches in the abstracts so we used a rather more limited query: ‘(title:software AND title:visuali* OR (keyword:software AND keyword:visuali*)).

C. Screening

After having queried the information sources using the search strings presented above and removed the duplicates, the relevant papers were selected using a set of inclusion/exclusion criteria. This was done in a screening phase, in which we checked the criteria on the title and abstract of each paper. Thus a paper was included if it:

- introduces an approach dealing with some aspects of software visualization and;
- presents an evaluation of the presented visualization approach.

A paper was excluded if:

- it does not deal with software visualization or it does not include an explicit evaluation,
- it does not present an abstract or its full text is not available,
- it consists of a compilation of work or tutorial in a conference or workshop,
- it is a short paper or keynote abstract.

Each candidate paper is screened by two members of the team to circumvent the subjective part of the inclusion/exlusion decisions. Papers having divergent decisions from the two members are discussed to reach a consensual decision. After screening all of the hundreds of papers, returned by the queries, by applying the inclusion/exclusion criteria, the number of papers to analyze was reduced considerably, which is always the case for systematic studies.

The full text of papers retained for the analysis were examined once more to ensure that they can be definitely included in the mapping study. Indeed, some of the papers, which were initially included because either their titles or their abstracts indicated the existence of an evaluation, were finally excluded after reading the article as no concrete evaluation was found. This further step slightly diminished the number of relevant papers.

The detailed results about the selection and the screening phases are given in Section III.

D. Classification Scheme

There are two schools of thought on how to determine the scheme for classifying the papers retained for the analysis. A first way, bottom-up, consists of extracting the classification scheme by reading the retained papers and by determining the

important classification properties. The second way, top-down, allows us to define the scheme using the general knowledge of the field. In our study, we used a hybrid approach, in which we combined our general knowledge with the information extracted from the abstracts during the screening phase.

We retained six classification properties:

- **Investigation Type**. This determines which type of empirical study was used: experiment, case study, or survey.
- **Tasks**. This specifies the nature of the tasks involved in the evaluation of the proposed visualization. Tasks are specific when a subject has to solve a specific problem. They are exploratory when the subject uses the visualization without a specific task to perform.
- **Data Sources**. This characterizes the source of the visualized data: industrial, open source and/or home data.
- **Subjects**. This determines the profile of subjects, if any, used in the evaluation: students, practitioners or both.
- **Measures**. This specifies if the evaluation included objective measures, i.e., not based on the subjects' judgement. Otherwise, the measures could be subjective or no measure was used for the assessment.
- **Comparison Reference**. If the proposed visualization tool is compared to other tools, this property allows us to specify the nature of the reference tools: visualization tools vs non-visualization tools performing the same task.

E. Data Extraction and Systematic Map

The actual mapping of relevant articles to categories of the classification scheme is performed during this phase. The reviewers analyze each article that was retained for the study and determine the appropriate class(es) for each. The frequencies of publications, and other metrics are calculated from the resulting classification. Section IV presents the results about the publications considered in this study.

III. Selection Process Results

A. Selection tool

To assist us in efficiently screening several hundred articles, we developed a lightweight tool. This tool behaves as a pipeline, with three successive features: setting, screening, and finalization.

1) Setting Feature: This feature parses the information exported by the search engines of IEEE Xplore and Scirus. Different parsing algorithms are used since the libraries export in different formats (*csv* for IEEE Xplore and *RIS* for Scirius). The parsing results are integrated together with the information for the additional manually-added papers, and the duplicates are detected and removed. Then the tool automatically assign the articles for the reviewers, by producing, for each of them, a *todo* file (recall that each articles has to be screened by two reviewers). The *todo* files contain the following data for each article:

$< Title > \$ < Authors > \$ < Abstract >$

2) Screening Feature: Using this feature, each reviewer enters his identifier and the script iterates over the entries in the *todo* file. The article is presented in a simple *Tk* window containing the authors, the title, the abstract in a scrolling text window, a checklist of rejection reasons and buttons to move to the next article or to save a session, as shown in Figure 2.

After screening each article, the decision is recorded in an output file and the next article is displayed. A screening session could be interrupted (*Save* option) and continued another time. When all the *todo* files are screened, we obtain one decision file per reviewer.

3) Finalization Feature: Three reviewers were involved in the screening process, and each of the papers was double evaluated by two reviewers. In this feature, the decision files of the three reviewers are merged. To classify the articles in three categories: *accepted* when both reviewers agree on acceptance, *rejected*, when both agree on rejection, and *Conflict* when one accept and the other reject the article.

To measure the degree of agreement between the reviewers, the tool calculates the Cohen's Kappa coefficient. Good values of Kappa indicate that the screening process is trustworthy, while bad values are a sign for ambiguous acceptance/rejection criteria.

Finally, in order to resolve conflicts found during the integration, face to face meetings were organized between the two reviewers to reach an agreement. If no agreement is reached, the third reviewer is involved to settle the conflict.

B. Selection results

Figure 3 shows the flow of information through the different phases of the study [7]. At each phase the number of articles is reduced by eliminating those that are not suitable for our study. The following paragraphs discuss this flow of information.

1) Sources:

a) Distribution: In total the search produced 773 candidate documents. The vast majority of the returned documents came from the *IEEE Xplore* database. This is explained by the fact that, in addition to IEEE publications, this database also indexes the publications of other publishers. Moreover, a large body of visualization work is published in conferences and journals sponsored by IEEE. The second largest set of articles came from the SoftVis series. Figure 4 illustrates the distribution of the publications over the three considered sources.

b) Duplicate Removal: An article exported automatically from both the IEEE Xplore and Scirus databases, is considered as duplicate. Our tool detected and removed 3 duplicated articles. Moreover, we had to remove 18 additional documents without authors (mainly conference descriptions. After these removals, 752 documents were screened from the initial 773 documents.

2) Screening Results:

a) Distribution of Rejections: After the screening phase, 103 documents were accepted for the analysis, which represent 14% of the documents. This is the usual acceptance rate in systematic literature/mapping studies. The 648 other

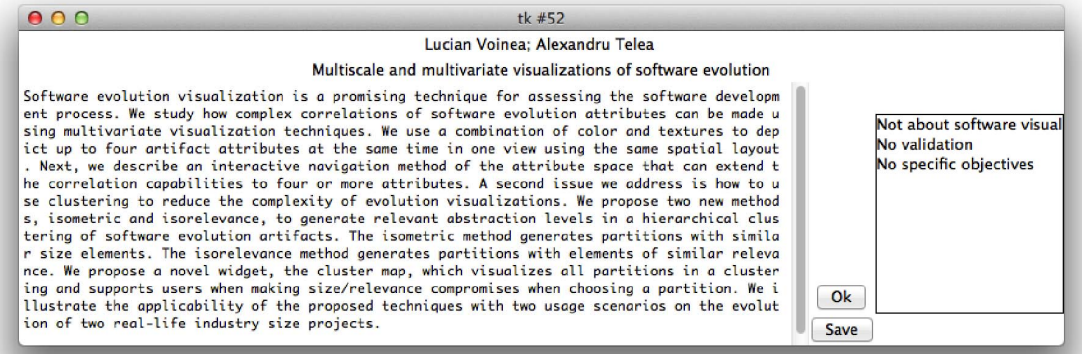

Fig. 2: Interface of the screening tool.

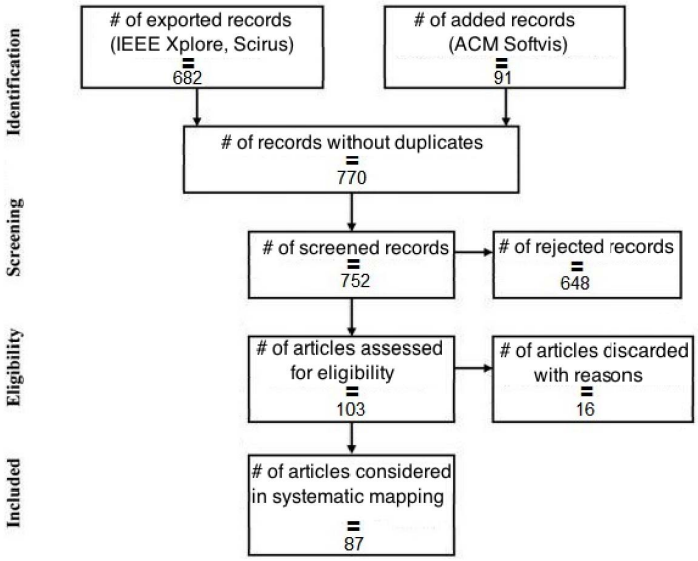

Fig. 3: Flow of information through our systematic mapping process [7].

documents were rejected for one of the three reasons as shown in Figure 5. By far, the primary cause of rejection during the screening was that the articles do not propose a specific software visualization contribution, although the titles and/or the abstracts contain the searched keywords. Several were rejected because both the title and the abstract do not mention any form of validation (validation keywords appear with another meaning). Finally, three articles were excluded because the validation refers simply to illustrative examples. During the screening, the reviewers scanned the whole articles when the titles and abstract were ambiguous about the contribution and/or the validation.

We only had a few articles with conflicting decisions. These conflicts were resolved by discussing our arguments and going rapidly over the whole article to remove any doubt.

b) Cohen's Kappa: We calculated the Cohen's Kappa between each pair of reviewers. We obtained an average Kappa of 0.44, which indicates a moderate agreement between the reviewers before the conflict resolution. Such a Kappa value indicates a low likelihood of random agreement [8].

IV. Study Results

For the purpose of our analysis, we classified the relevant papers according to the properties defined in Section II. This classification aims at answering the research questions formulated in the same section. At this stage, it is worth mentioning that 16 articles were excluded during the classification and after reading the full text. The main reason for the rejection was that, although the abstracts mentioned that there is a validation, no serious validation was described. Thus, the following results concern the remaining 87 papers. The

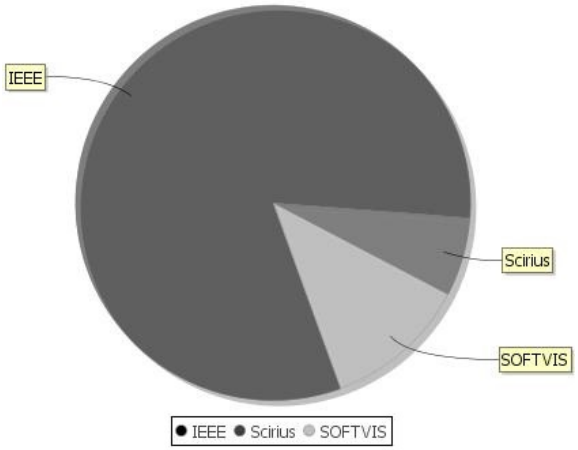

Fig. 4: Distribution of Sources

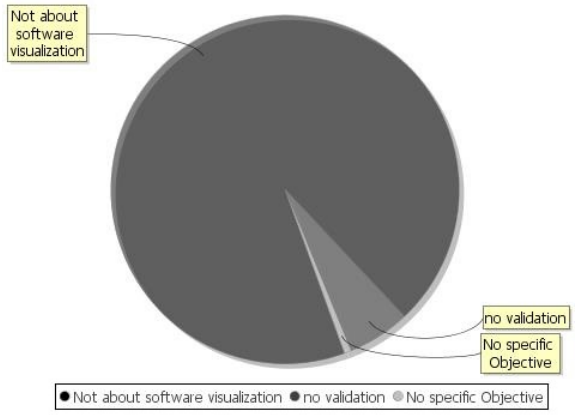

Fig. 5: Distribution of article rejections by reason

following sub-sections present the results for each property of the classification scheme. We also present the distributions of the analyzed papers over the time, publishers, and publication venues.

A. Investigation Types

We consider the three usual investigation strategies.

1) Experiment: It is used when it is possible to control the studied phenomenon, that is, to manipulate the factors which can influence it, to validate or invalidate a hypothesis. 16.09% of reviewed papers report on an experiment. The studied hypotheses consist of testing particular cause and effect relationships (e.g., the proposed visualization tool helps decrease the comprehension effort). As usual, the studies try to reject a null hypothesis (e.g., there is no difference in comprehension effort using the proposed visualization tool). The null hypothesis is rejected in favor of the alternative hypothesis when the p-value of the statistical test used is less than a given threshold (usually 0.05). As an example, Murphy-Hill et al. [9] proposed a visualization tool to detect code smells. They evaluate their visualization by conducting an experiment in which they test several hypotheses about the proposed visualization. For example, they confirmed, with a Wilcoxon test, the hypothesis that *programmers identify more smells using the tool than not using the tool*. Another example of validation using an experiment is reported in [10]. The authors successfully compared the use of their tool with the one of Eclipse in terms of correctness and completion time of comprehension tasks. Their null hypotheses were rejected using a two-ways ANOVA test.

2) Case study: Less controlled than an experiment, a case study consists in studying a given case (product, project, organization) to collect detailed information or to make an exploration. Its outcome often serves as preliminary information of an experiment. The accepted articles are, for the vast majority, case studies (78.16 %). These case studies feature roughly 65% purely qualitative analysis, which is rather high. Most of them aim at evaluating visualizations using user feedback on the effort spent using the tools. The quantitative analysis tends to have similar objectives but using one or more metrics. Moreover, in many cases, these metrics are used for comparison purposes. The work of Islam et al. [11] is an example of a qualitative evaluation using two case studies. The authors evaluated the ability of their heat-map based visualization to analyze two programs in order understand their structure and the interaction between their components.

3) Survey: It is a retrospective study of a situation; it can be done in description purposes, such as seeing what the distribution of certain characteristics in a population is, or to explain certain choices, or exploratory for preparation for more thorough study. Only 5 surveys were found in the analyzed articles. This is rather surprising given that surveys seem to be the de facto standard when attempting to measure the user's opinion on subjective matters like usefulness of a given visualization. Most articles either went for case studies, controlled experiments or simply used heuristic and self-assessment to determine tool usability. A paper that presents a survey is [12] in which the type of investigation mentioned in the abstract is a case study. However, after the analysis, we classified the investigation as a survey because the used procedure matches the survey definition. Indeed, the authors conducted interviews, based on an inial informative survey, to figure out how developers understand exception-handling constructs and whether the proposed visualization tool can assist them.

It is worth noting that the majority of experiments and surveys are used in SOFTVIS papers.

B. Tasks

As mentioned earlier, the tasks used in validation studies can be either specific or exploratory.

More than two third of the articles involve specific tasks in the evaluation (72.5%). In the majority of the cases, the tasks are described as part of the study setting. For example, Murphy-Hill et al. [9] define the specific tasks *identifying smells in code* and *making refactoring judgements*.

The remaining 27.5% articles do not use specific tasks. Either the subjects use the evaluated tool in free-exploration mode or the authors discuss their visualization without addressing a specific task. An example of the last case is [13], in which the author shows that the proposed visualization is possible and practical for real applications.

C. Data source

When conducting tool evaluations, there are three types of data that could be used: open source, industrial and home data. As expected, 77% (67 of 87) of the articles used open source projects. Java projects are the most used. For example, Azureus and Freemind are studied in Langelier et al. [14]. Other languages are considered, for example C++ in [15] where the wxWidgets code base, a popular C++ GUI library, is utilized. Industrial projects are the second larger category of data involved in the validations (23%). This is pretty good considering the difficulty of obtaining this category of data (confidentiality, and brand image issues). Another argument in favor of using open source data is the ease of comparing visualization tools as the category of data is easy to exchange between the researchers in the field than the industrial one. A good example of using industrial data is provided in [16]. The authors use BRec, a tool for reconstructing 3D building models from laser scan data, to evaluate their visualization of multithreaded execution traces. Finally, only 8% of the analyzed articles conducted evaluations with home data. This small proportion can be explained by the fact that using such a kind of data will not produce the same level of confidence on the results as the one for industrial and well-known open-source projects.

D. Subjects

Visualization involves, by definition, human actors. Therefore, we expect that validations involve in some way subjects. Actually, a large majority of evaluations are done without subjects (70.1%). The usual scenario, as in [14] or [16], is to apply the tool on data and discuss the resulting views on anecdotal situations. Sometimes the tool is compared to others using equivalent views of the same situations.

When subjects are involved, their profile is important because it influences the generalization of the conclusions. Validations with practitioners are in general more suitable than those with students. However, this comes at a cost. For this reason, students are often used. Indeed, 13.8% of the considered papers propose a validation only with student subjects (see, for example, [17]). By comparison, validations with only practitioners represents 6.9% of the analyzed articles (see, for example, [18]). The remaining articles (9.2%) use both students and practitioners as in [9]. In many cases, the experience of the subjects is considered as a mitigating factor as in [19].

E. Measure

One aspect that can impact the validity of an evaluation (experiment or case study) is the nature of the used measures.

Property	Class	Number of articles
Investigation Type	Experiment	14
	Case Study	68
	Survey	5
Task	Specific	63
	Exploratory	24
Data Source	Open Source	67
	Industrial	20
	Home Data	7
Subjects	None	61
	Students	12
	Practitioners	6
	Both	8
Objective Measures	Yes	53
	No	34
Comparison	None	67
	Visualization	14
	Domain	2
	Both	4

TABLE I: Category Distribution.

We distinguish between two possibilities in our analysis: (i) the use of objective measures (e.g., time to perform a task, number of defects detected by the subject, etc.), (ii) and no use of objective measures. The measure could be subjective (e.g., effort level as perceived by the subjects, usefulness of the tool on a Likert scale, etc.), or no measures in the case of qualitative studies. Our analysis revealed that a majority of the articles (60.9%) collect objective measures. For example, Deng et al. [20] compare their Constellation tool with SeeSoft in terms of lines of code to be explored. The other articles (39.1%) use subjective measures or just do not collect measures.

F. Comparison

As for any form of research, authors of software visualization articles have to demonstrate that their contribution advances the state of the art or the practice. This is generally done by comparing the proposed contribution with existing ones. Surprisingly, 77% of the articles did not include any comparison to other approaches. This high proportion can be explained by the variety of input formats and data used by the tools. The availability of the state of the art tool is also an issue. When comparisons are performed, these involve only other visualization tools in 16% of the cases, only non-visualization tools used to solve the same tasks in 2.3% of the cases, or both types of tools in 4.6% of the cases. An example of comparison between visualization tools can be seen in [20] (Constellation vs SeeSoft). The other form of comparison is illustrated in [10] where CodeCity is compared to the combination of Eclipse and Excel.

G. Yearly Distribution

The year to year distribution shows that the number of papers increased after 2005 (Figure 6). After this date, the

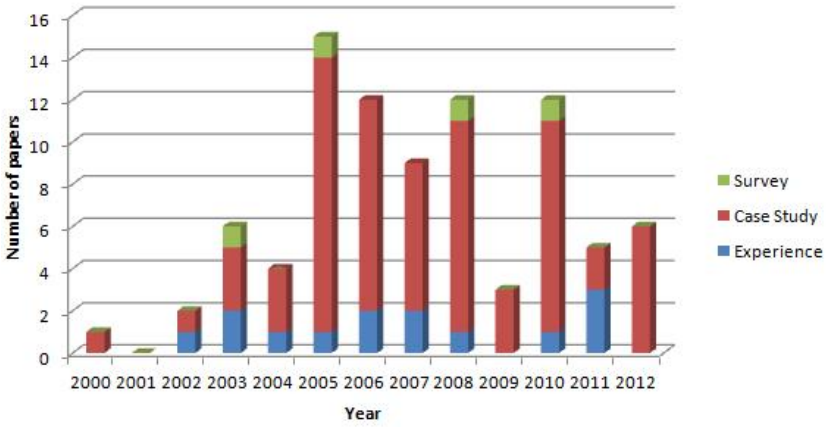

Fig. 6: Investigation type distribution per year.

Publisher	Number of articles
IEEE	47
Elsevier North-Holland	1
ACM	35
Cépaduès	1
European Association for Computer Graphics	3

TABLE II: Distribution of articles per publisher.

Venue Category	Number of articles
Software visualization	48
Maintenance and evolution	12
Graphics and visualization	11
Software engineering	11
Other computer science	5

TABLE III: Distribution of studies per venue category.

years with few validated papers are those when SoftVis was not organized. The distribution mode corresponds to the collocation of SoftVis with ICSE. Collocation with VL/HCC (2008) and the VisWeek, also showed an increase in the validated articles.

H. Publisher Distribution

Despite the bias in articles selection against ACM publications, due to the absence of automatic exports, we find that the large majority of articles that passed through the screening process were published either by IEEE or ACM (see Table II). This is not really surprising since these publishers host the largest conferences in the domain. However, we noticed that this study's acceptance rate for other publishers was extremely low.

I. Publication Venue Distribution

Table III gives an overview of the studies according to publication venue categories. We distinguish between these categories according to their target communities. The most represented one in terms of the number of articles is, without surprise, the software visualization community (principally Softvis and VISSOFT). It accounts for 55.2% of the analyzed articles. The second largest category includes maintenance and evolution venues such as ICPC, WCRE, CSMR (13.79%). Here again, this was expected as most of the tasks addressed by the software visualization research cover the maintenance phase. Similar proportions (12.6%) concern the categories of

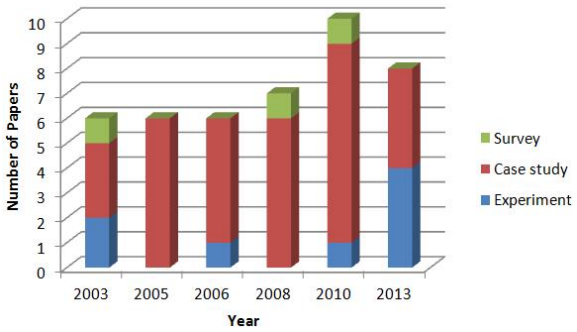

Fig. 7: Softvis investigation type distribution per year.

general software engineering conferences like ICSE, and general graphics and visualization conferences such as EUROVIS and InfoVis. Finally, 5.75% of the articles reporting validations come from other computer science venues.

V. SoftVis specific results

After the global analysis, we studied specifically the SoftVis conference publications. In addition to the SoftVis articles (2003 to 2010) retained during the systematic mapping process (see Figure 3), we added the articles published in the VISSOFT 2013 conference since SoftVis merged with VISSOFT that year. The results of this specific study are summarized in TableIV.

Much like the more general study, the majority of the

Property	Class	Number of articles
Investigation Type	Experiment	8
	Case Study	32
	Survey	3
Task	Specific	34
	Exploratory	9
Data Source	Open Source	36
	Industrial	8
	Home Data	5
Subjects	None	27
	Students	11
	Practitioners	3
	Both	2
Objective Measure	Yes	26
	No	17
Comparison	None	36
	Visualization	4
	Domain	3
	Both	0

TABLE IV: Softvis category distribution

SoftVis articles used the case studies as investigation type (see, for example, [21], [22], [23], [24]). In 2013, articles using experiments as the investigation type account for 50% of the validated articles, which is exceptional. For example, Sharif et al. [19] assess their visualization tool SeeIT 3D using an experiment. Only 7% of the articles used surveys to evaluate the proposed visualizations, such as the work reported in [25]. Figure 7 presents the distribution of the articles and their investigation types over the years.

For the task nature, many articles targeted specific tasks when validating their claims. In fact, 79.1% of the SoftVis articles include specific tasks during validation, which is slightly higher than the score of the general study.

In the context of the third classification property, the data used is extracted mostly from open source projects (83.7%) as it is the case in [19]. For 18.6% of the articles, industrial projects were involved as in [26], which is slightly lower in proportion than in the general study. We conjecture that when industrial data is available, the authors tend to submit their papers to more prestigious conferences and journals. Finally, home data was more used in SoftVis that in the general study (11.6%). Note that an article can describe more than one source of data, which explains the total percentage over 100%.

With respect to subjects, 62.8% of the articles do not report use of human subjects in their evaluation (see, for example, [24]). When the validation includes human subjects, 11 of the 16 articles involved only use student subjects, which confirms our belief about the ease of hiring students subjects compared to practitioners. Only 3 evaluations were done with only practitioners as in [27]. Finally, 2 articles used a combination of students and practitioners.

The proportion of articles using objective measures is almost

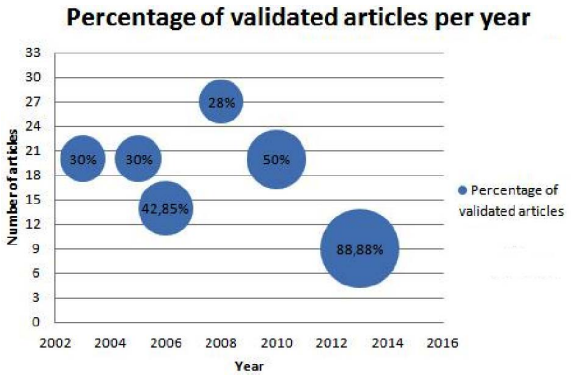

Fig. 8: Ratio of validated articles per year.

the same as in the general study (60.46%). For example, the time to localize a fault was used as a measure in [28]. Finally, the comparisons performed during the evaluations are scarce in the SoftVis series (16.27%) compared to the general study (23%).

As we have all the articles published in SoftVis, we looked at the evolution of validated paper over the years. This evolution is shown in Figure 8. The higher percentages in the last two editions (2010 and 2013) suggest an increasing maturity in the research validation in the software visualization community. However, except for 2013, the number of rigorous controlled experiments is still low, and the majority of validations are qualitative case studies without subjects.

VI. Study Limitations

As with any systematic study, many considerations can limit the validity of the drawn conclusions. In this section, we discuss the most important ones.

A. Selection Bias

The major limitation of our process is the absence of an exhaustive search. We selected databases that allow us to export the results, which eliminates potential sources such as ACM Digital Library. For example, some conferences were published alternatively by IEEE and ACM and thus, we missed potential papers during the ACM years. A good example is the paper [29] of Automated Software Engineering, published by ACM in 2005. Still, we ensured that major venues in software visualization were included in the study. Moreover, even if our search is not exhaustive, we considered a large sample of the body of work. Another limitation of the selection is the systematic rejection of short papers with the assumption that the limited number of pages does not allow to describe a validation. We noticed afterwards that some of these papers propose rigorous validations and have a high number of citations (see, for example, [30]).

B. Format Errors

During the screening process, we experienced some text formatting issues with our tool. We corrected these issues, but some of the articles were rated with the first version. Since

we were working in parallel but at different moments, some of these articles have been rated with the two versions (each reviewer using a different version). This generated potential errors in the integration step. To mitigate this issue, we have manually inspected the ratings performed with the two versions to correct the integration.

C. Screening

One issue with the screening process arose from the fact that, considering the hundreds of papers returned by the search, we can only afford to consult the abstract of articles. Consequently, we may have rejected articles that had a poor abstract but valid content. To try to mitigate this issue, we used two reviewers for each article and consulted the whole content to settle any divergent opinion on a paper. Still, it is probable that an article that does not contain a reference to validation in the abstract or title will be excluded. We have tried to address this issue by instructing the reviewers to include the paper if there is a reasonable possibility that a validation is provided.

D. Classification

We prevented classification errors by a careful definition of classes and a cross check by a second person as described in section III. Nevertheless, there are some borderline cases where an article could be classified in more than one category. For example, in the article [12], the authors present as validation of their claim a case study but we decided to classify the article as a survey after analysis and discussion between the reviewers. During the evaluation task, the reviewers decided how to classify the articles assigned to them. Despite the expertise of the reviewers, some articles are difficult to classify due to the unclear boundaries between some classification scheme categories and also their presentation in those articles.

VII. Related Works

As our work concerned the systematic mapping of validation strategies of software visualization research, the related work of interest pertains to the mapping studies, the validation reported in software engineering literature and the surveys of software visualization.

Software engineering has become a mature field. Consequently, the number of articles analyzing the state of the art in this discipline has noticeably increased. The studies conducted in these articles vary in methodology and goals but aim to summarize a certain aspect of the research done in software engineering. A methodology used with increasing frequency is the systematic literature review. These studies, such as the one in [4], target a specific topic and perform an in-depth analysis of the published empirical results on this topic. Another type of study is the systematic mapping which analyzes the trends in publications in a field of interest. For instance, in [31], the authors investigate the testing of software product lines. We perform a systematic mapping to highlight the general directions in software visualization validation.

There are many contributions studying the evaluation and validation of work done in the software engineering community. Tichy et al [32] performed a study about the experimental evaluation in computer science in general. The authors compared the computer science research to other fields with regard to the use of experimental evaluation to validate the research work. Zelkowitz and Wallace [33] developed a 12-step taxonomy for classifying experimental validation. They classified 612 papers in the software engineering literature and concluded that the taxonomy gives good results. Furthermore, they observed that the software engineering community performs poorly when it comes to validating its claims. Zelkowitz [34] updated the previous survey by including recent years publications. Sjoeberg et al. [35] conducted a survey of controlled experiments in software engineering. They investigated 5453 articles published in 12 leading software engineering journals and conferences from 1993 to 2002. They quantitatively characterize the experiments performed and concluded that a low proportion of software engineering articles reports controlled experiments. All these studies review the validation techniques in software engineering in general. The main outcome that is shared by these studies is the poor results of software engineering in the evaluation and validation reported in the published articles. Our systematic mapping in the software visualization validation confirms these findings in this specific sub-domain of software engineering.

Software visualization domain has been constantly growing this past decade and has reached a certain maturity. Although we did not find systematic mapping studies targeting software visualization, there are some contributions studying the state of art of this field. Caserta and Zendra [36] present a survey of the visualization of static aspects of software. They reviewed and summarized visualization techniques and categorized them according to their features and characteristics. Cornelissen et al. [37] present a systematic review of dynamic program comprehension tools, a category including a large segment of software visualization's application. They have reviewed papers from the largest conferences of software visualization, but approach this analysis from a different perspective. Papers on software visualization are not singled out from other comprehension tools and they did not consider papers on visualization that were not using dynamic analysis. More specific to the field, Sensalire et al. [38] performed a comparative analysis of four studies on software visualization tools. Finally, Kienle and Muller [39] carried out a comprehensive literature survey in order to outline seven requirements on software visualization tools in order to guide future effort. These articles provide comprehensive views on software visualization tools. We concentrate on the evaluation reported in the literature.

VIII. Conclusion

In this paper, we report on a systematic mapping study of validation of software visualization research. Our study uses two online databases, IEEE Xplore and Scirus, along with selected articles from SOFTVIS conference. This study, which covers the period 2000-2012 was conducted following the systematic mapping process. First, we collected all publications found by querying the databases. Then, we screened them using their abstract to ensure that they were eligible for our

analysis. Finally, we analyzed every included article to classify the validation according to our scheme.

We found that publications with validation were fairly constant, year to year, since 2003 with peaks years in 2005, 2006, 2008 and 2010. Analysis of the types of experiment done shows a clear bias toward case studies. Roughly, half of the articles used quantitative measures. Few used subjects as part of the evaluation. Clearly, the domain still has some way to go to reach the necessary maturity in validating its claims. The specific analysis of SoftVis publications did not reveal major gaps in the general study. It would be interesting to further analyze trends in publication validation to identify the causes of the shortage in rigorous validations such as controlled experiments, the use of objective measures and human subjects' evaluation.

REFERENCES

[1] S. Diehl, *Software Visualization: Visualizing the Structure, Behaviour, and Evolution of Software*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2007.

[2] K. Petersen, R. Feldt, S. Mujtaba, and M. Mattsson, "Systematic mapping studies in software engineering," in *Proc. of the 12th Int. Conf. on Eval. and Asses. in Soft. Eng.*, ser. EASE'08. Swinton, UK, UK: British Computer Society, 2008.

[3] B. Kitchenham and S. Charters, "Guidelines for performing Systematic Literature Reviews in Software Engineering," Keele University and Durham University Joint Report, Tech. Rep., 2007.

[4] B. Kitchenham, R. Pretorius, D. Budgen, O. Pearl Brereton, M. Turner, M. Niazi, and S. Linkman, "Systematic literature reviews in software engineering - a tertiary study," *Inf. Softw. Technol.*, Aug. 2010.

[5] F. Q. B. da Silva, A. L. M. Santos, S. C. B. Soares, A. C. C. França, and C. V. F. Monteiro, "A critical appraisal of systematic reviews in software engineering from the perspective of the research questions asked in the reviews," in *Proc. of the 2010 ACM-IEEE Int. Symp. on Emp.l Soft. Eng. and Measur.*, ser. ESEM '10. New York, NY, USA: ACM, 2010.

[6] O. Barbosa and C. Alves, "A systematic mapping study on software ecosystems." in *IWSECO@ICSOB*, ser. CEUR Workshop Proceedings, S. Jansen, J. Bosch, P. R. J. Campbell, and F. Ahmed, Eds., vol. 746. CEUR-WS.org, 2011, pp. 15–26.

[7] D. Moher, A. Liberati, J. Tetzlaff, and D. Altman, "Preferred reporting items for systematic reviews and meta-analyses: the prisma statement." *Annals of Internal Medicine*, vol. 151, no. 5, pp. 264–9, W64, 2009.

[8] J. R. Landis and G. G. Koch, "The measurement of observer agreement for categorical data," *Biometrics*, vol. 33, no. 1, pp. 159–174, 1977.

[9] E. Murphy-Hill and A. P. Black, "An interactive ambient visualization for code smells," in *Proc. of the 5th Int. Symp. on Soft. Vis.*, ser. SOFTVIS '10. New York, NY, USA: ACM, 2010.

[10] R. Wettel, M. Lanza, and R. Robbes, "Software systems as cities: a controlled experiment," in *Proc. Int. Conf. on Soft. Eng.*, 2011, pp. 551–560.

[11] S. S. Islam, J. Krinke, and D. Binkley, "Dependence cluster visualization," in *Proc. of the 5th Int. Symp. on Soft. Vis.*, ser. SOFTVIS '10, 2010.

[12] H. Shah, C. Görg, and M. J. Harrold, "Visualization of exception handling constructs to support program understanding," in *Proc. of the 4th ACM Symp. on Soft. Vis.*, ser. SoftVis '08. New York, NY, USA: ACM, 2008.

[13] S. P. Reiss, "Visualizing java in action," in *Proc. of the 2003 ACM Symp. on Soft. Vis.*, ser. SoftVis '03. New York, NY, USA: ACM, 2003.

[14] G. Langelier, H. Sahraoui, and P. Poulin, "Exploring the evolution of software quality with animated visualization," in *IEEE Symp. on Vis. Lang. and Human-Centric Comp.*, 2008.

[15] A. Telea and L. Voinea, "An interactive reverse engineering environment for large-scale c++ code," in *Proc. of the 4th ACM Symp. on Soft. Vis.*, ser. SoftVis '08. New York, NY, USA: ACM, 2008.

[16] J. Trümper, J. Bohnet, and J. Döllner, "Understanding complex multi-threaded software systems by using trace visualization," in *Proc. of the 5th Int Symp. on Soft. Vis.*, ser. SOFTVIS '10. New York, NY, USA: ACM, 2010.

[17] J. Sajaniemi and M. Kuittinen, "Program animation based on the roles of variables," in *Proc. of the 2003 ACM Symp. on Soft. Vis.*, ser. SoftVis '03. New York, NY, USA: ACM, 2003.

[18] J. Lawrence, S. Clarke, M. Burnett, and G. Rothermel, "How well do professional developers test with code coverage visualizations? an empirical study," in *Vis. Lang. and Human-Cent. Comp., 2005 IEEE Symp. on*, Sept 2005, pp. 53–60.

[19] B. Sharif, G. Jetty, J. Aponte, and E. Parra, "An empirical study assessing the effect of seeit 3d on comprehension," in *Soft. Vis. (VISSOFT), 2013 First IEEE Work. Conf. on*, Sept 2013.

[20] F. Deng, N. DiGiuseppe, and J. A. Jones, "Constellation visualization: Augmenting program dependence with dynamic information." in *VISSOFT*. IEEE, 2011.

[21] O. Benomar, H. Sahraoui, and P. Poulin, "Visualizing software dynamicities with heat maps," in *Soft. Vis. (VISSOFT), 2013 First IEEE Work. Conf. on*, Sept 2013.

[22] M. Pinzger, H. Gall, M. Fischer, and M. Lanza, "Visualizing multiple evolution metrics," in *Proc. of the 2005 ACM Symp. on Soft. Vis.*, ser. SoftVis '05. New York, NY, USA: ACM, 2005.

[23] M. Sensalire, P. Ogao, and A. Telea, "Classifying desirable features of software visualization tools for corrective maintenance," in *Proc. of the 4th ACM Symp. on Soft. Vis.*, ser. SoftVis '08. New York, NY, USA: ACM, 2008.

[24] R. Lintern, J. Michaud, M.-A. Storey, and X. Wu, "Plugging-in visualization: Experiences integrating a visualization tool with eclipse," in *Proc. of the 2003 ACM Symp. on Soft. Vis.*, ser. SoftVis '03. New York, NY, USA: ACM, 2003.

[25] C. Anslow, S. Marshall, J. Noble, E. Tempero, and R. Biddle, "User evaluation of polymetric views using a large visualization wall," in *Proc. of the 5th Int. Symp. on Soft. Vis.*, ser. SOFTVIS '10. New York, NY, USA: ACM, 2010.

[26] W. De Pauw and S. Heisig, "Zinsight: A visual and analytic environment for exploring large event traces," in *Proc. of the 5th Int. Symp. on Soft. Vis.*, ser. SOFTVIS '10. New York, NY, USA: ACM, 2010.

[27] A. Abuthawabeh, F. Beck, D. Zeckzer, and S. Diehl, "Finding structures in multi-type code couplings with node-link and matrix visualizations," in *Soft. Vis. (VISSOFT), 2013 First IEEE Work. Conf. on*, 2013.

[28] C. Gouveia, J. Campos, and R. Abreu, "Using html5 visualizations in software fault localization," in *Soft. Vis. (VISSOFT), 2013 First IEEE Work. Conf. on*, Sept 2013.

[29] G. Langelier, H. Sahraoui, and P. Poulin, "Visualization-based analysis of quality for large-scale software systems," in *International Conf. on Auto. Soft. Eng.*, 2005.

[30] K. Dhambri, H. Sahraoui, and P. Poulin, "Visual detection of design anomalies," in *Eur. Conf. on Soft. Maint. and Reeng.*, 2008.

[31] P. A. da Mota Silveira Neto, I. do Carmo Machado, J. D. McGregor, E. S. de Almeida, and S. R. de Lemos Meira, "A systematic mapping study of software product lines testing," *Information and Software Technology*, vol. 53, no. 5, pp. 407 – 423, 2011.

[32] W. F. Tichy, P. Lukowicz, L. Prechelt, and E. A. Heinz, "Experimental evaluation in computer science: A quantitative study," *Journal of Systems and Software*, vol. 28, no. 1, pp. 9 – 18, 1995.

[33] M. V. Zelkowitz and D. Wallace, "Experimental validation in software engineering," *Information and Software Technology*, vol. 39, no. 11, 1997.

[34] M. V. Zelkowitz, "An update to experimental models for validating computer technology," *Journal of Systems and Software*, vol. 82, 2009.

[35] D. Sjoeberg, J. Hannay, O. Hansen, V. Kampenes, A. Karahasanovic, N.-K. Liborg, and A. Rekdal, "A survey of controlled experiments in software engineering," *IEEE Trans. on Soft. Eng.*, vol. 31, no. 9, 2005.

[36] P. Caserta and O. Zendra, "Visualization of the static aspects of software: A survey." *IEEE Trans. Vis. Comput. Graph.*, vol. 17, no. 7, pp. 913–933, 2011.

[37] B. Cornelissen, A. Zaidman, A. van Deursen, L. Moonen, and R. Koschke, "A systematic survey of program comprehension through dynamic analysis," *Soft. Eng., IEEE Trans. on*, vol. 35, no. 5, pp. 684–702, Sept 2009.

[38] M. Sensalire, P. Ogao, and A. Telea, "Evaluation of software visualization tools: Lessons learned," in *Vis. Soft. for Underst. and Analysis, 2009. 5th IEEE Int. Work. on*, 2009.

[39] H. Kienle and H. Muller, "Requirements of software visualization tools: A literature survey," in *Vis. Soft. for Underst. and Analysis, 2007. 4th IEEE Int. Work. on*, June 2007.