Information and Software Technology 71 (2016) 77–91

Contents lists available at ScienceDirect

Information and Software Technology

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

Domain-Specific Languages: A Systematic Mapping Study

Tomaž Kosar[a,*], Sudev Bohra[b], Marjan Mernik[a]

[a] University of Maribor, Faculty of Electrical Engineering and Computer Science, Smetanova ulica 17, 2000 Maribor, Slovenia
[b] Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213-3890, USA

ARTICLE INFO

Article history:
Received 22 December 2014
Revised 24 September 2015
Accepted 10 November 2015
Available online 21 November 2015

Keywords:
Domain-Specific Languages
Systematic Mapping Study
Systematic Review

ABSTRACT

Context: In this study we report on a Systematic Mapping Study (SMS) for Domain-Specific Languages (DSLs), based on an automatic search including primary studies from journals, conferences, and workshops during the period from 2006 until 2012.

Objective: The main objective of the described work was to perform an SMS on DSLs to better understand the DSL research field, identify research trends, and any possible open issues. The set of research questions was inspired by a DSL survey paper published in 2005.

Method: We conducted a SMS over 5 stages: defining research questions, conducting the search, screening, classifying, and data extraction. Our SMS included 1153 candidate primary studies from the ISI Web of Science and ACM Digital Library, 390 primary studies were classified after screening.

Results: This SMS discusses two main research questions: research space and trends/demographics of the literature within the field of DSLs. Both research questions are further subdivided into several research sub-questions. The results from the first research question clearly show that the DSL community focuses more on the development of new techniques/methods rather than investigating the integrations of DSLs with other software engineering processes or measuring the effectiveness of DSL approaches. Furthermore, there is a clear lack of evaluation research. Amongst different DSL development phases more attention is needed in regard to domain analysis, validation, and maintenance. The second research question revealed that the number of publications remains stable, and has not increased over the years. Top cited papers and venues are mentioned, as well as identifying the more active institutions carrying DSL research.

Conclusion: The statistical findings regarding research questions paint an interesting picture about the mainstreams of the DSL community, as well as open issues where researchers can improve their research in their future work.

1. Introduction

"*Domain-specific languages (DSLs) are languages tailored to a specific application domain. They offer substantial gains in expressiveness and ease of use compared with general-purpose programming languages in their domain of application [45].*" As such, DSLs [16,21,27,30,37,45,47,54] become an emerging popular area of research within the field of Software Engineering (SE), and one of the more important constituents of software development methodologies such as: Generative Programming [14], Product Lines [63], Software Factories [26], Language-Oriented Programming [62], and Model-Driven Engineering (MDE) [25,56–58]. On the other hand, research on DSLs has never become a truly independent research field with established research groups and long-lasting conferences/workshops. Despite some attempts to consolidate DSL research groups by organising USENIX Conferences on Domain-Specific Languages in 1997 and 1999, series of HICSS Minitracks on Domain-Specific Languages for Software Engineering in 2001–2003, IFIP Working Conference on Domain-Specific Languages in 2009 and 2011, and more recently SPLASH 2014 workshop on Domain-Specific Language Design and Implementation. More often DSL researchers have published their works either under broader communities such as programming language research (e.g., Symposium on Principles of Programming Languages; Conference on Programming Language Design and Implementation; Conference on Systems, Programming, Languages and Applications: Software for Humanity; Conference on Functional Programming; Symposium on Practical Aspects of Declarative Languages), or within specific application domains for which DSLs were developed (e.g., embedded systems, high-performance computing, electronic commerce, robotics). Furthermore, DSLs can

* Corresponding author. Tel.: +386 22207448.
E-mail address: tomaz.kosar@um.si (T. Kosar).

http://dx.doi.org/10.1016/j.infsof.2015.11.001
0950-5849/

be developed in more varied ways than General-Purpose Languages (GPLs). For example, during the design phase a new DSL can be based on already existing language (language exploitation pattern [45]), or designed from scratch without any relationship to an existing language (language invention pattern [45]). Whilst, independently from a design phase a DSL can be implemented by different approaches (e.g., interpreter, compiler, preprocessing, embedding, extensible compiler/interpreter, COTS, hybrid [45]), each having its own merits [36]. Due to the fact that research on DSLs is spreading into many software development methodologies, vast areas of application domains, and different development approaches, it is hard to obtain a complete knowledge of the DSL research field, and foreseen DSL research trends. Therefore, the main objective of the described work was to perform a Systematic Mapping Study (SMS) [31,52] on DSLs for better understanding the DSL research field, identifying research trends, and possible open issues.

Note that the term 'domain-specific language' has also been used within the MDE community as a synonym for a 'domain-specific modeling language' (DSML). On one hand, the differences between DSLs and DSMLs are not unbridgeable and many similarities in the designs and implementations of DSLs and DSMLs can be identified. As well as dissimilarities between the syntax descriptions used (grammars vs. metamodels) [50], and the semantic description immaturities of DSMLs compared to DSLs [9]. Thus, due to the aforementioned persisting differences between DSLs and DSMLs researchers have recently become more careful when using both terms. On the other hand modeling communities, and in particular domain-specific modeling communities, rarely cooperate with a programming language community. They have established their own set of conferences (e.g., Conference on Model Driven Engineering Languages and Systems; Domain-Specific Modeling Workshop) and journals (e.g., Journal of Software and Systems Modeling). Although, one of the goal of the Software Language Engineering (SLE) [35] series of conferences is to bring both communities together. SLE is a young engineering discipline with the aim of establishing systematic and rigorous approaches to the development, use, and maintenance of computer languages. However, SLE comprises general-purpose and domain-specific specification, modeling, and programming languages. Hence, its scope is much broader than DSLs. In this work we concentrated solely on grammar-based DSLs, which are intrinsically textual. We plan to perform similar SMS on DSMLs as future work. The main contributions of this paper are:

- an overview of DSL research since the DSL survey paper [45] was published 10 years ago identifying trends and gaps in DSL research,
- better classification on primary studies in the DSL research than presented in [49] by deriving more accurate conclusions,
- applying SMS to a broader field of DSLs, thus enhancing its usability,
- improving best practices on performing SMSs (e.g., keywording process would be better if replaced by taking into account survey papers within a research field and/or by contacting experts within that research field), and
- enhancing reliabilities of SMSs (e.g., classification of empirical or non-empirical research).

This paper is organised as follows. Related work on SMSs in general and within DSLs are discussed in Section 2. Description of research method, SMS planning and execution details are highlighted in Section 3. In Section 4, we report the results of performed SMS according to the research questions defined in Section 3. A discussion on threats to validity is presented in Section 5. Key findings and concluding remarks of SMS on DSLs with an outline for future work are summarised in Section 6.

2. Related work

A systematic review (SR) is a secondary study that reviews primary studies with the aim of synthesising evidence related to a specific research question. Several forms of SRs exists [31], depending on the depth of reviewing the primary studies (e.g., performing quality assessment of the primary studies), and on the specificities of research questions:

- Systematic literature review (SLR): "*A form of secondary study that uses a well-defined methodology to identify, analyse and interpret all available evidence related to a specific research question in a way that is unbiased and (to a degree) repeatable [31].*"
- Systematic mapping study (SMS): "*A broad review of primary studies in a specific topic area that aims to identify what evidence is available on the topic [31].*"
- Tertiary review (TR): "*which is a systematic review of systematic reviews [31].*"

Hence, SLRs are more driven by specific research questions (e.g., is one particular approach better than other), whilst research questions in SMS are of a higher-level (e.g., which empirical methods have been used, which research topics have been addressed). A further guideline as to whether to perform a SLR or a SMS is that the latter is more appropriate if it is discovered that very little evidence is likely to exist or that the topic is very broad [31]. Due to the broadness of DSLs, as discussed in Section 1, the emphasis in this work is on SMS for DSLs. A more detailed definition of SMS can be found in [52]: "*The main goal of systematic mapping studies is to provide an overview of a research area, and identifing the quantity and type of research and results available within it. Often one wants to map the frequencies of publication over time to see trends. A secondary goal can be to identify the forums in which research in the area has been published.*"

It is important to point out that both SLRs and SMSs have established rigorous methodologies [31] for performing such studies. Although, that methodology for SR in SE has recently attracted more attention, this young SE methodology still suffers from some infancy problems. Many of them are stated in [10,32,34,66]:

- usefulness of many SMSs is not as was expected (e.g., SMSs have not as yet been used by other researchers as a starting point for their research, SMSs benefit more researchers than practitioners),
- problems of classifying studies in a replicative manner, and
- classification of primary studies by inexperienced researchers.

Overall, we still need to obtain more experience in performing various SMSs by acquiring knowledge on how to perform and use them. Hopefully, this study will introduce an additional facet of knowledge towards gaining more experience with such studies.

By examining the literature on existing SMSs we noticed that many of them were performed on research topics with very little existing evidence. In an extreme case as little as 13 primary studies were identified and examined [5]. There is a clear trend in current SMSs towards selecting a research topic which is not too broad in nature. One possible reason might be that a SMS that needs to examine several hundred primary studies is even more time-consuming and challenging. On the other hand the usefulness of such SMSs is severely hampered due the narrowness of the research topic. We are convinced that SMSs would be of much greater use if they were applied to broader research topics. However, this previous claim can only be proved after such SMSs have indeed been performed on broader topics. This is also the aim of SMS in this paper. The only SMS we found with more than 1000 examined and classified primary studies was SMS on DSLs [49] but, this SMS [49] revealed most of the problems of current SMSs as previously mentioned and discussed in [10,66]. In particular, we didn't find that study [49] very useful as the authors classify the primary studies regarding a research focus with respect to keywords found in the primary studies and not to already-established

research focus within the DSL field. Although the authors of [49] follow the guidelines on how to perform SMSs in SE [52] the outcome classification of research focus (in the authors' words a DSL research type) on ADL (Architecture Description Language), DSAL (Domain-Specific Aspect Language), DSML, external DSL, internal DSL, method or process, technique, and tools that are far from being satisfactory. Our classification is presented in Section 3. The problem of classifying the primary studies based on the keywords found in the primary studies (keywording method) has already been identified in [10,55]. Our proposal to overcome a problem, unfamiliarity of researchers performing such studies within the research field under discussion, is to apply SMSs either when the categories are known in advance (e.g., ACM classification can be used as suggested in [66], existing taxonomies from survey papers), or experts from the field under investigation are consulted and asked for help during categorisation. Note, that whilst conducting this SMS and simultaneously writing this paper we indeed found a SMS [13] where classification of test case prioritisation was already based on an existing survey paper [67]. Hence, the keywording method has now already been replaced in some SMSs by existing taxonomies from survey papers, which is in our opinion good practice.

As the existing SMS on DSLs [49] is unsatisfactory we decided to repeat it. This was not an exact replication of SMS in [49] due to differences in research questions and classifications, as well as the inclusion of primary studies (we concentrated solely on DSLs, and DSMLs were excluded). However, we should point out that SMS on DSLs [49] also portrayed some application domains where DSLs and DSMLs have been applied. About 30 different application domains were identified and 15 more frequent application domains were mentioned in [49] (e.g., control systems, data intensive apps, embedded systems, security, simulation, testing, web). However, within our SMS application domain identification was not a part of the research question.

Whilst completing our SMS another SLR on DSLs recently appeared for investigating DSL type systems (see Chapter 2 in [42]). However, there are differences in the research questions between our SMS and the SLR in [42]. In the latter case the research questions were completely oriented towards type systems in DSLs (e.g., How are DSL type systems described? How complex are DSL type systems? Are DSL type systems static? Are DSL type systems strong?), whilst our research questions were much broader (see Section 3).

Amongst different SRs recently performed we would like to point out the work of [66]. This is interesting and important work as the authors compared two different SMSs ([18] and [48]), which were performed by two independent research groups on the same topic – software product line testing. The main finding from the study [66] is that the reliability of SMSs should not simply be taken for granted and that a "*standardized classification scheme with an agreed interpretation*" [66] is need. As a response to this quest we suggest two level classifications.

3. Research method

We performed SMS on DSLs based on the guidelines presented in [31,52], using good practices from previous similar SMSs (e.g., [1,4,8,18,20,23,40,43]), and based on very recent findings from comparing the same SMSs performed by two independent research groups [66].The protocol is available at [39].

The following simplified structure for performing SMS has been suggested in [52] and was used in our study:

- defining research questions,
- conducting a search for primary studies,
- screening primary studies based on inclusion/exclusion criteria,
- classifying the primary studies, and
- data extraction and aggregation.

3.1. Defining research questions

The objective of this study was to obtain a comprehensive overview of current DSL research. The following research questions were defined for elaborating on this overall goal.

*RQ*1: What has been the research space of the literature within the field of DSLs since the survey paper on DSLs [45] was published 10 years ago?

*RQ*2: What have been the trends and demographics of the literature within the field of DSLs after the survey on DSLs [45] was published 10 years ago?

In a very broad sense we were interested in DSL studies from three different prospectives: type of contribution, type of research, and focus area. It was for these reasons that the research question *RQ*1 was further split into three sub-questions.

*RQ*1.1 *Type of contribution*: What is the main contribution of DSL studies with respect to techniques/methods, tools, processes, and measurements?

Answering *RQ*1.1 would enable us to assess whether the DSL community is more focused on developing new techniques/methods for particular DSL development phases, or on developing new tools for DSL development, or on DSL processes, or on DSL measurements.

*RQ*1.2 *Type of research*: What types of research methods have been used in DSL studies?

Answering *RQ*1.2 would allow us to assess maturity within the field (e.g., whether DSL researchers have been using empirical or non-empirical research methods; whether they are using controlled experiments).

*RQ*1.3 *Focus area*: Which research topics have been investigated in DSL studies?

The following specific focus areas based on a DSL survey paper [45] have been included in this SMS:

- DSL development phases. Which DSL development phase was described in a study? Answering this question would allow us to identify those DSL development phases (domain analysis, design, implementation, validation, maintenance) that are currently underrepresented.
- DSL domain analysis. Was domain analysis performed formally or informally? If former which of the formal domain analysis method was used (e.g., FODA [29], FAST [63])? Answering this question would allow us to identify whether informal domain analysis prevailed over formal domain analysis.
- DSL design phase. In which manner were the DSLs designed (two orthogonal dimensions: formal vs. informal/language exploitation vs. language invention [45])? Answering this question would allow us to assess whether internal DSLs prevailed over external DSLs [21] (internal DSL is roughly equivalent to language exploitation; external DSL is roughly equivalent to language invention).
- DSL implementation phase. In which manner the DSLs were implemented? Answering this question would allow us to identify which implementation pattern [45] has been the more often used (interpreter, compiler, preprocessor, embedding, extensible compiler/interpreter, COTS, hybrid). In the case of internal DSLs an interesting question would be which GPLs (e.g., Ruby, Haskell, Scala) have been used for DSL embedding? What has been the trend over the years? Answering this question would allow us to identify the more popular GPLs used for embedding. Has there been any change in popularity recently (e.g., has Haskell been replaced by Scala)? Which DSL development tools/frameworks have been used for external DSL implementations (e.g., ANTLR [51], Stratego/XT [7], LISA [44,46])? Which tools/frameworks have been the more popular?
- DSL validation phase. Has the DSL design been validated by end-users?

- DSL maintenance phase. Has DSL been evolving? Is there any experience about DSL maintenance/evolution?

The research question *RQ2*, which was about the trends and demographics of DSL research, was further split into four sub-questions.

- *RQ2.1 Publication count by year*: What has been the annual number of publications within this field?
- *RQ2.2 Top cited papers*: Which DSL primary studies used in this SMS have been cited the most?
- *RQ2.3 Active institutions*: Rather than identifying researchers within the DSL field we opted for identifying DSL groups working at particular institutions. This was measured by the number of published papers. How DSL groups are connected together (being co-authors of a same primary study)?
- *RQ2.4 Top venues*: Which venues (e.g., journals, conferences, workshops) have been the main targets of DSL papers? Note, that there have been no specialised DSL journals or conferences spanning many years. Hence, it would be interesting to know where DSL researchers have been mostly published.

Many papers on SMSs have followed the guidelines [52] for classifying the types of research methods defined in [64]. Often, those easy to understand and easy to use are mentioned [34] as a reason for using this classification. However, it has only recently been shown in [66] that this might not be so. Two or more independent research groups might classify the same studies differently. For example, only 11 out of 33 studies were classified equally in [18] and [48] with the consequences of different conclusions. The studies disagreed as to whether validation research or evaluation research was predominant [66]. Despite this threat we are still convinced that the aforementioned classification is simple and useful. However, in our SMS some improvements to the process of classification have been incorporated within the aim of being a more reliable and replicable process. In [52] it was suggested that the types of research methods could be further classified into empirical research (validation research and evaluation research), and into non-empirical research (opinion paper, experience paper, philosophical/conceptual paper, and solution proposal). However, it seems that this coarse-grained classification has been unacceptable amongst SR researchers so far. In our opinion, this broader classification is very useful for obtaining a broader picture of the research field and is more reliable than the fine-grained classification. For example, the disagreement between two studies ([18] and [48]) would be smaller regarding whether during the product line testing more empirical/non-empirical research were performed. However, there would still be disagreements between [18] and [48] because some primary studies have been classified as evaluation research by one research group and as a solution proposal by another research group. Hence, such different classifications (note, that inside the group common agreement was reached amongst several researchers) might be avoided in the future if classification covers two levels. In the first level the study would be classified as whether it is empirical or non-empirical research, and only later more fine-grained classification would be done. A refinement of the classification has been proposed very recently in [53] with more detailed guidelines for classification, which can be seen as an additional effort towards a standardised classification scheme with an agreed interpretation [66].

3.2. Search conducting, papers screening, and classification

3.2.1. Conducting the search for primary studies

The search string and inclusion/exclusion criteria were defined in order to address the research questions *RQ1* and *RQ2*. Finally, by following the protocol [39] all relevant primary studies were classified within the selected Digital Libraries (DLs).

Table 1
Preliminary identification of relevant publications.

Digital library	Accessible at	No. of publications
ISI web of science	http://sub3.webofknowledge.com	792
ACM digital library	http://dl.acm.org	361
		Σ 1153

The following elementary search string has been used in our SMS (the rationale is described in more detail in the protocol [39]:

("domain-specific language" OR "DSL")

AND year $>$ 2005 *AND year* $<$ 2013

It is interesting to compare our search string with those used in [49]:

"*domain-specific language*" *OR*

"*domain-specific modeling language*" *OR*

"*generative programming*"

Within the search string of [49] there were no constrains on the years of publishing and were also interested on DSMLs. Amongst different software development methodologies where DSLs play an important role (see Section 1) they only opted for Generative Programming. In our SMS the interest is not on particular software development methodology, nor on DSMLs. Hence, these terms have been excluded within the search string.

The search string was applied on the following set of DLs (Table 1) after following the protocol [39].

3.2.2. Screening papers based on inclusion/exclusion criteria

In our study the following inclusion criteria were used:

- study must have addressed DSL research,
- peer reviewed studies had been published in journals, conferences, and workshops,
- study must be written in English,
- study must be accessible electronically, and
- computer science literature.

The exclusion criteria were:

- irrelevant publications that lay outside the core DSL research field, which also excluded DSMLs, modelware, and MDE publications, visual/graphical languages (based on graph-grammars or other formalisms) or those mentioning DSL as future work;
- non-peer reviewed studies (abstracts, tutorials, editorials, slides, talks, tool demonstrations, posters, panels, keynotes, technical reports);
- peer-reviewed but not published in journals, conferences, workshops (e.g., PhD thesis, books, patents);
- publications not in English;
- electronically non-accessible; and
- non-computer science literature.

The inclusion and exclusion criteria were applied to the titles, keywords, and abstracts. In those case where it wasn't completely clear from the title, keywords, and abstract that a publication really addressed the DSL research then such publications were temporarily included but might be excluded during the next phase (classification phase) when the whole publication (not only the abstract) had been read. Hence, only publications that were clearly outside the scope were excluded during this phase. After the screening of 1153 publications (see Table 1) 713 publications satisfied the aforementioned

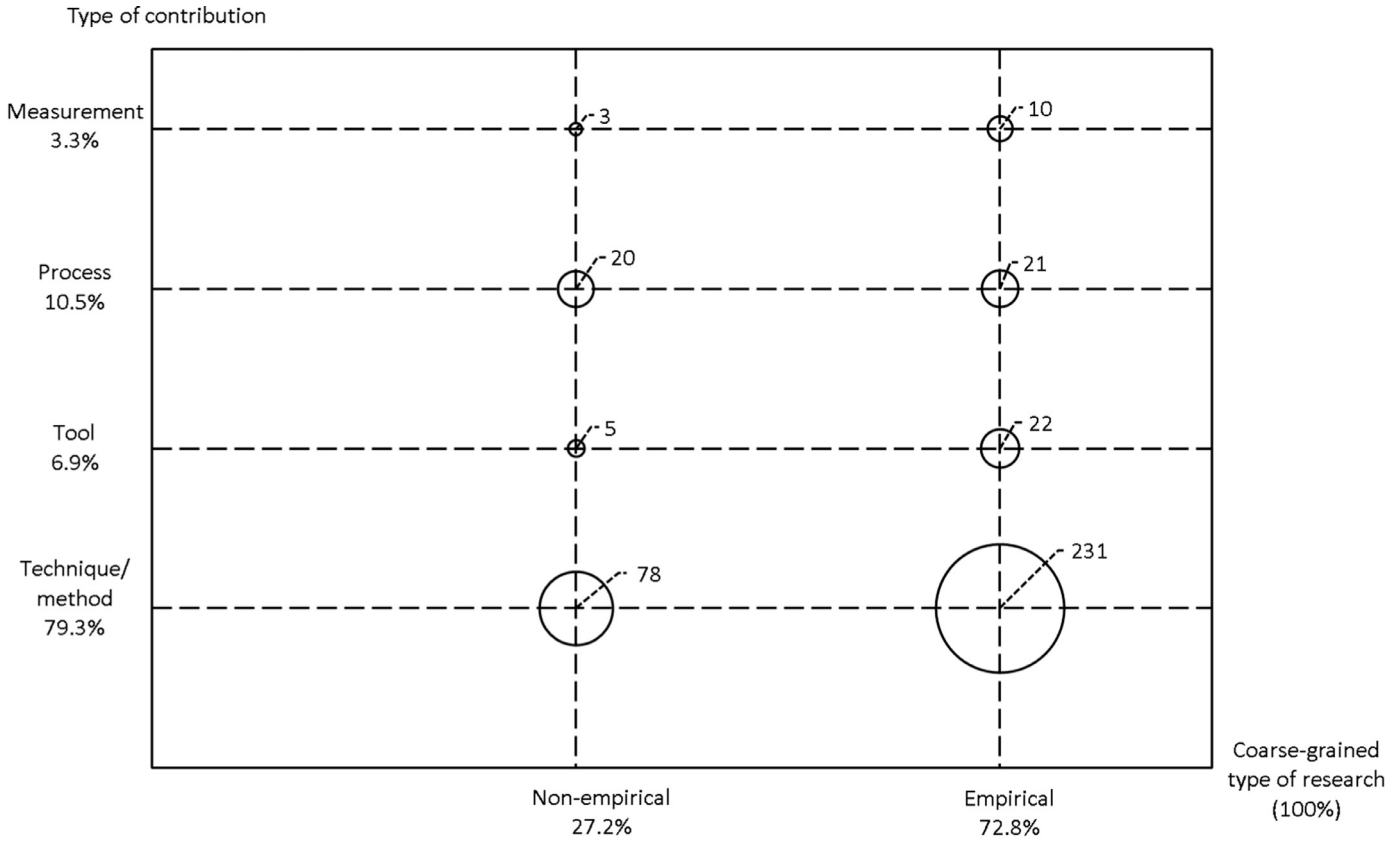

Fig. 1. Number of primary studies per type of contribution (RQ 1.1) and coarse-grained type of research (non-empirical vs. empirical) (RQ 1.2), $n = 390$.

criteria and entered into the next phase – classification, where an additional 323 publications were then excluded. Hence, altogether 390 primary studies were classified.

3.2.3. Classifying the papers

Classification is one of the more critical and time-consuming steps when conducting SMSs. In regard to our research questions, it was not expected that all relevant information could only be inferred from the abstracts. Hence, the classification was done based on reading the whole primary study.

It was recommended [31] that the classification of primary studies is done by at least two authors, and in the case of disagreement the authors need to decide how to resolve it. However, such a process only increases the reliability of classification inside the group, where probably the opinions of the more senior researchers would prevail anyway. It was shown in [66] that this process didn't improve the reliabilities of classifications between different research groups. To recall, in the case of two independent SMSs on product line testing 22 out of 33 papers were classified differently (see Section 3.1 and [66]). Due to these reasons our approach was slightly different. In our study classification of the primary studies was done by a single and the most experienced researcher (by the number of relevant DSL publications). In such a manner we also eliminated one of the identified shortcomings of SMS – classification of primary studies by inexperienced researchers [10]. Note that there are also some other SRs where classifications had been done by a single researcher (e.g., [33]). During this phase similar studies were also identified (e.g., if a conference paper had a more recent journal version we include only the latter). It was for this reason that we started classifying backwards, from 2012 to 2006. In order to avoid a fatigue during classification (as a possible threat to validity) classification was performed in blocks of at most two hours followed by at least one hour breaks.

4. Data extraction and aggregation: overview of the results

This section presents and discusses the results from classifying 390 primary studies. The objective of this study was to provide an overview of DSL research after the DSL survey paper [45] had been published. The following research questions were identified (see Section 3):

- *RQ*1.1 *Type of contribution*
- *RQ*1.2 *Type of research*
- *RQ*1.3 *Focus area*
- *RQ*2.1 *Publication count by year*
- *RQ*2.2 *Top cited papers*
- *RQ*2.3 *Active institutions*
- *RQ*2.4 *Top venues*

Fig. 1 shows (y-axes) an overview on the type of contribution. It can be observed that the DSL community has been more interested in developing new techniques/methods (79.3%) that supported different DSL development phases, rather than investing in developing new DSL tools (6.9%). Primary studies about integration of DSLs into other SE processes have also been rare (10.5%), whilst studies about measuring the effectiveness of DSL approaches have been almost non-existent (3.3%). Indeed new DSL tools have rarely been developed over the past 10 years (an example of a new tool is Neverlang [61]). The matured tools (e.g., ANTLR [51], Stratego/XT [7], see also Fig. 6) have merely been enhanced with new features. Recent comparisons between DSL tools have been discussed in [19]. A small percentage (6.9%) should not be of great concern. However, the situation is probably quite different for DSML where the research field has not yet reached the level of DSL maturity and many DSML tools have been developed recently. Interestingly, a DSML tool Xtext [17] has been used

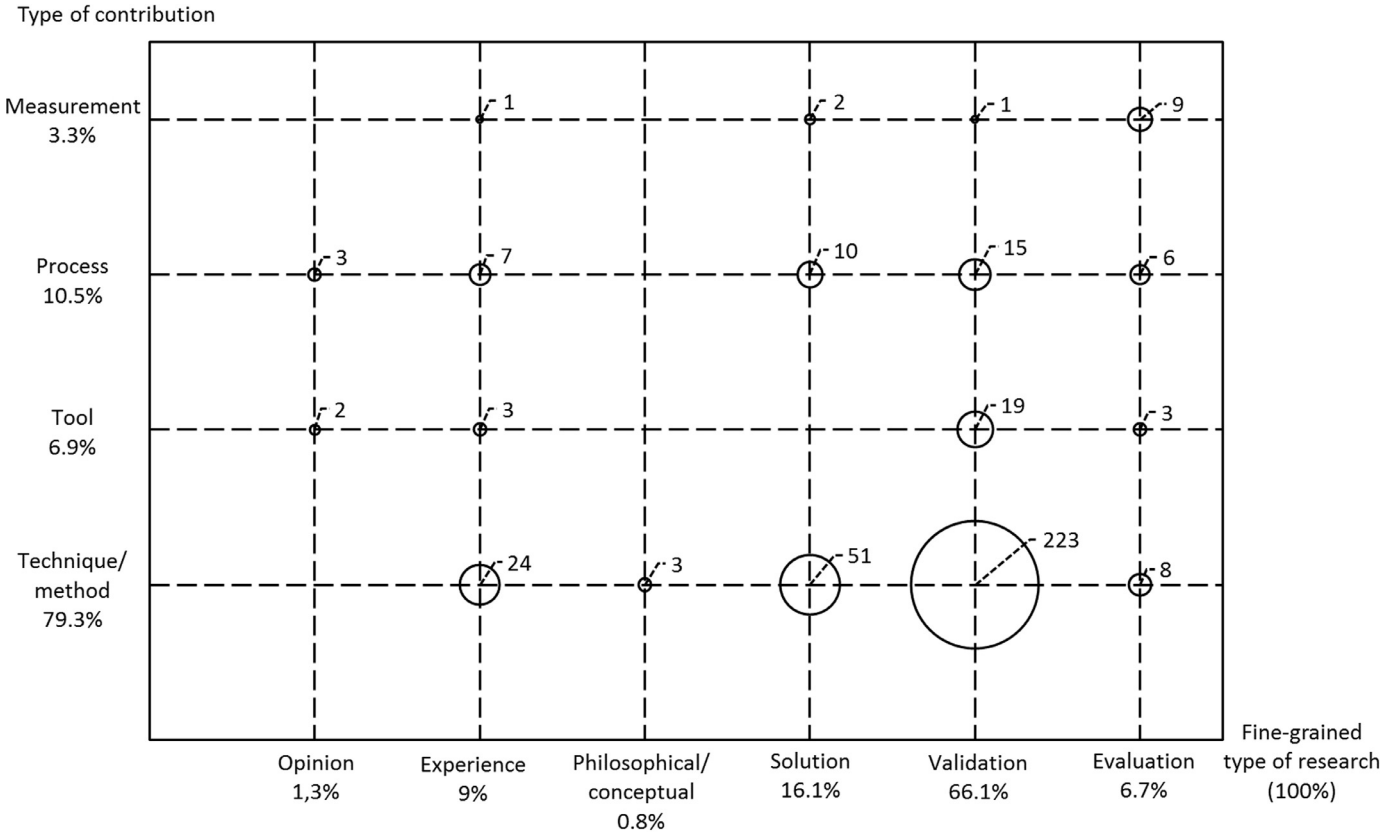

Fig. 2. Number of primary studies per type of contribution (RQ 1.1) and fine-grained type of research (RQ 1.2), $n = 390$.

for grammar-based DSLs as well. However, to fully support this claim another SMS should be carried out on DSMLs, which would be our future work. On the other hand, it is quite clear that there is a lack of DSL research regarding processes and measurements. More emphasis should be placed in the near future on integrating DSLs into other SE processes, as well as about measuring the effectiveness of DSL approaches.

It can also be observed from Fig. 1 (x-axes) that empirical research (72.8%) has prevailed over non-empirical research (27.2%). This is an indication of the maturity of DSL research. It can be observed from Fig.2 that opinion and philosophical/conceptual primary studies have been rare (about 2.1%) in DSL research. The same is true for experience reports (9%). It can also be concluded that the presented ideas have also been implemented at least at the level of a prototype and hence only solution proposals have been rare (16.1%). Whilst the ratio between empirical and non-empirical research (72.8% vs. 27.2%) has been much higher than within some other research fields (e.g., 18% of empirical research has been reported for aspect-oriented model-driven code generation in [43], 28% of empirical research for software quality trade-off in [4], 34% of empirical research for software product line testing in [18]) and hence satisfying, but this can't be claimed for the ratio between validation and evaluation DSL research (66.1% vs. 6.7%). There has been a clear lack of evaluation research into all types of contribution (technique/method, tool, process, measurement) (Fig. 2). A need for empirical evaluation in software modeling research is discussed in [12], where it was found that the rigour of empirically-validated research has been rather weak. In particular, the authors found only 4% of controlled experiments within the research works published during the period 2006–2010 at conferences on Model Driven Engineering Languages and Systems. This number is even slightly higher than the number we found in this paper (1.3% of controlled experiments) and further indicates, as in [12], that researchers within the DSL community are more interested in creating new techniques than they are in performing rigorous empirical evaluations (e.g., [38]).

Fig. 3 shows which research topics (focus area) have been investigated in DSL research classified into various DSL development phases. Note that a particular primary study can include various development phases. Hence, the total number was higher than the sum of included primary studies ($n = 390$). It is obvious from Fig. 3 that the primary studies usually discussed the following three DSL development phases: domain analysis, design and implementation, whilst validation and maintenance have been rarely presented. Indeed, in many primary studies we found a brief section on domain analysis identifying the main concepts of DSL under development followed by the design of DSL syntax and semantics and finalising with implementation details. On the other hand, there has been a lack of DSL research about domain analysis (only 1.3% of primary studies have concentrated solely on domain analysis). Of particular concern should be the lack of DSL research within the validation phase. DSLs had rarely been validated (e.g., by end-users) assuming that the developed DSLs were perfectly tailored for domains, as well as fitting end-users requirements [22]. However, this is far from true. DSL under development should have been validated by empirical studies, involvement of end-users, or by the psychology of programming research. Recent attempts in this direction are works [2,3,11,60].

Although, that DSL primary studies mostly included the domain analysis (229/390 ≈ 58.7%), design (291/390 ≈ 74.6%) and implementation (280/390 ≈ 71.8%) phases, it doesn't mean that researchers have followed the best practices of a particular phase. It can be observed from Table 2 that only 5.7% of primary studies that included domain analysis had used a formal domain analysis approach. Hence, formal domain analysis methods had been rarely used in DSL development and domain analysis had usually been done informally and probably in an incomplete manner. The rarity of formal methods within the domain analysis phase is further exhibited in Fig. 4

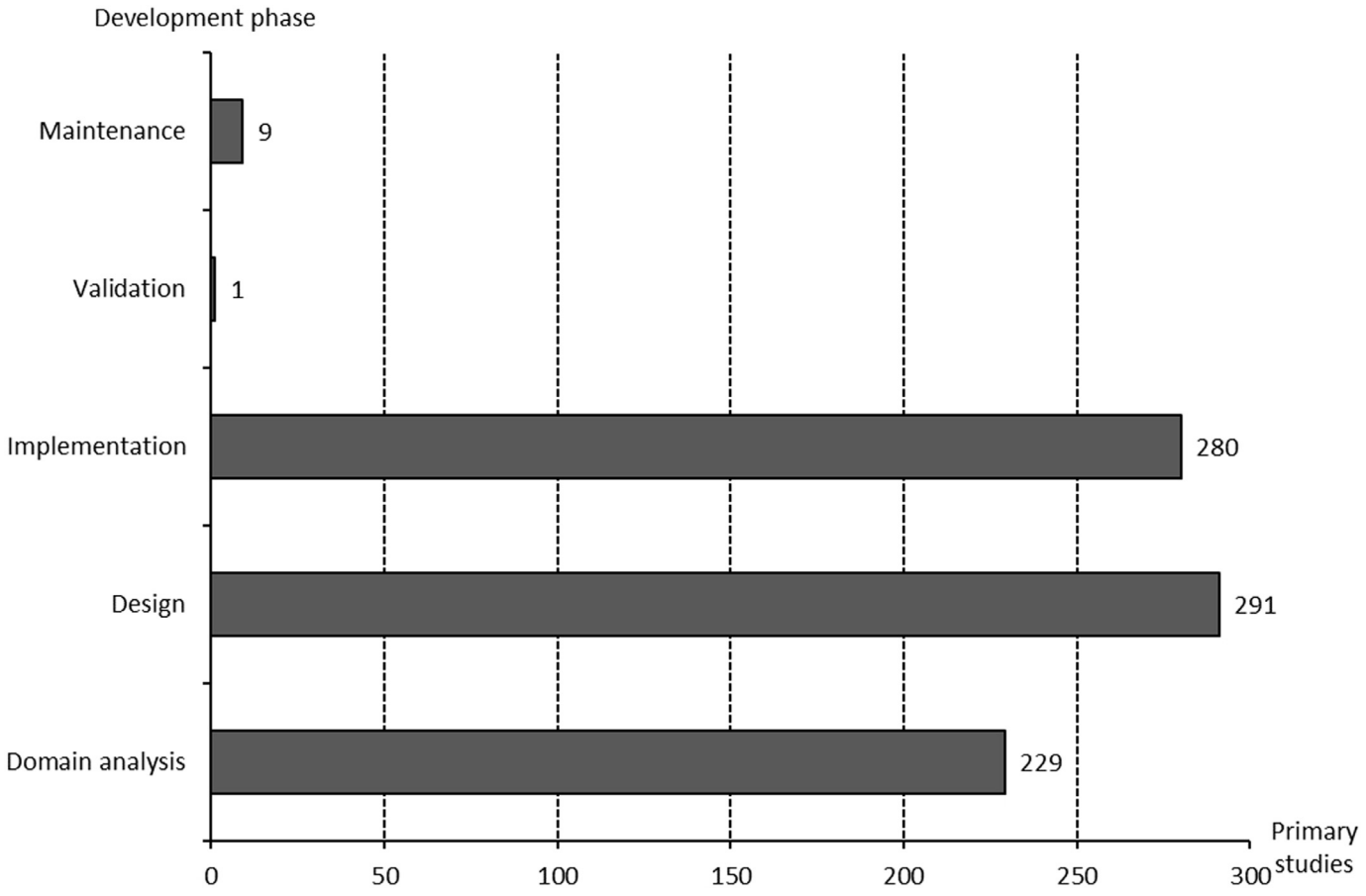

Fig. 3. Research distribution in DSL development phases (RQ 1.3).

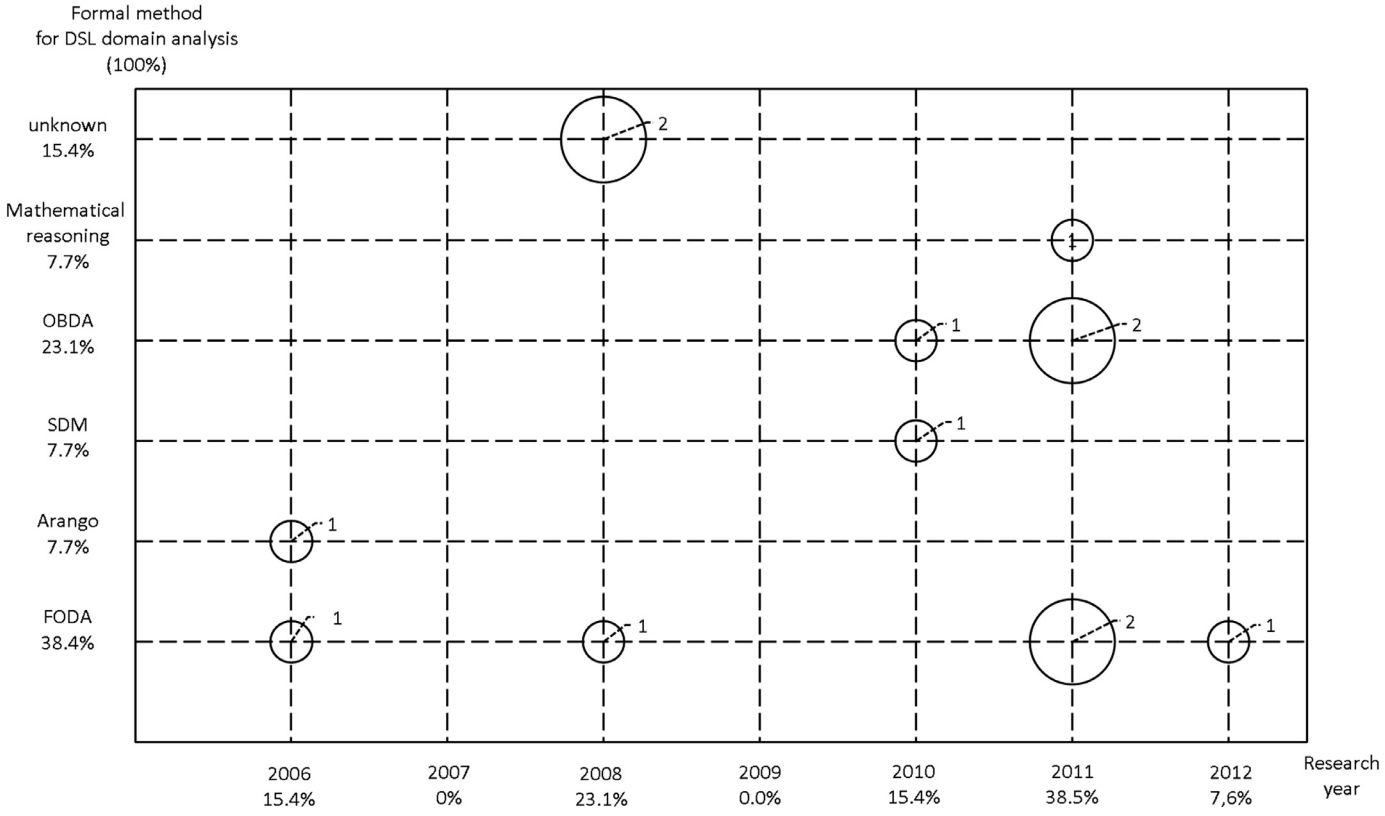

Fig. 4. Distribution of formal methods used in domain analysis per years (RQ 1.3), $n_2 = 13$.

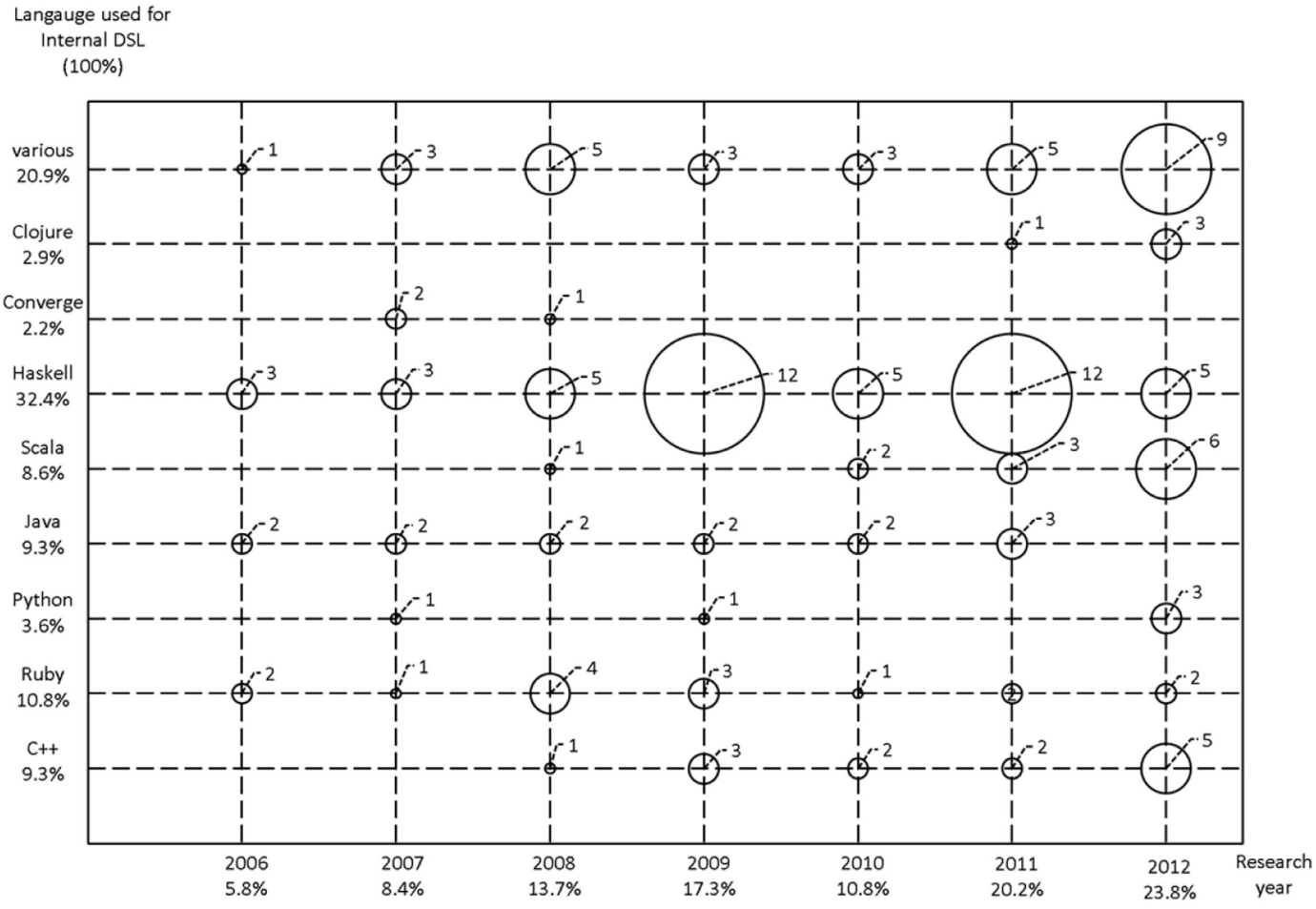

Fig. 5. Distribution of GPLs used for DSL embedding per years (RQ 1.3), $n_4 = 139$.

Table 2
Research distribution in DSL domain analysis (RQ 1.3), $n_1 = 229$.

Approach	Percentage (%)
Informal	94.3
Formal	5.7

Table 3
Research distribution in DSL design (informal vs. formal) (RQ 1.3), $n_3 = 291$.

Approach	Percentage (%)
Informal	83.2
Formal	16.8

Table 4
Research distribution of DSL design (internal vs. external) (RQ 1.3), $n_3 = 291$.

Approach	Percentage(%)
Internal	47.8
External	52.2

showing the distribution per years, as well as the more frequently FODA [29] had been used in domain analysis. There is an urgent need in DSL research for identifying the reasons for lack of using formal methods within domain analysis and possible solutions for improvement. The first observation might be that information gathered during domain analysis cannot be automatically used in the language design process. Another reason might also be that complete domain analysis is too complex and outside of software engineers' capabilities. DSL researchers should look into available domain analysis tool and investigate how they can be accommodated for supporting the DSL domain analysis phase. As noted in [41]: "*The execution of a domain analysis process without a tool support can lead to an unsuccessful result, due to the complexity of interrelated activities and work products.*" Furthermore, only 1.3% of primary studies used in our SMS concentrated solely on the domain analysis phase. Investigating the domain analysis phase has clearly been insufficient. Recent attempt in this direction is work [15].

A slightly better, although still unsatisfactory, situation than within the domain analysis phase is revealed in Table 3 where it can be observed that only 16.8% of primary studies that included the design phase used formal approaches for describing syntax and semantics. Although internal DSLs, which rarely require formal description as they rely on (formal) description of existing language, comprised 47.8% (Table 4) out of 83.2% informal cases, the number of DSLs using formal syntax and especially semantic description had still been low. Again, the DSL community should identify the reasons and works towards improvement.

Figs. 5 and 6 reveal some interesting information about internal and external DSLs [21]. It can be observed from Fig. 5 that Haskell has still been the more frequently used host language. Whilst from Fig. 6 (category unknown) it can be observed that the existing tools had been rarely used (or not reported in the primary studies) for

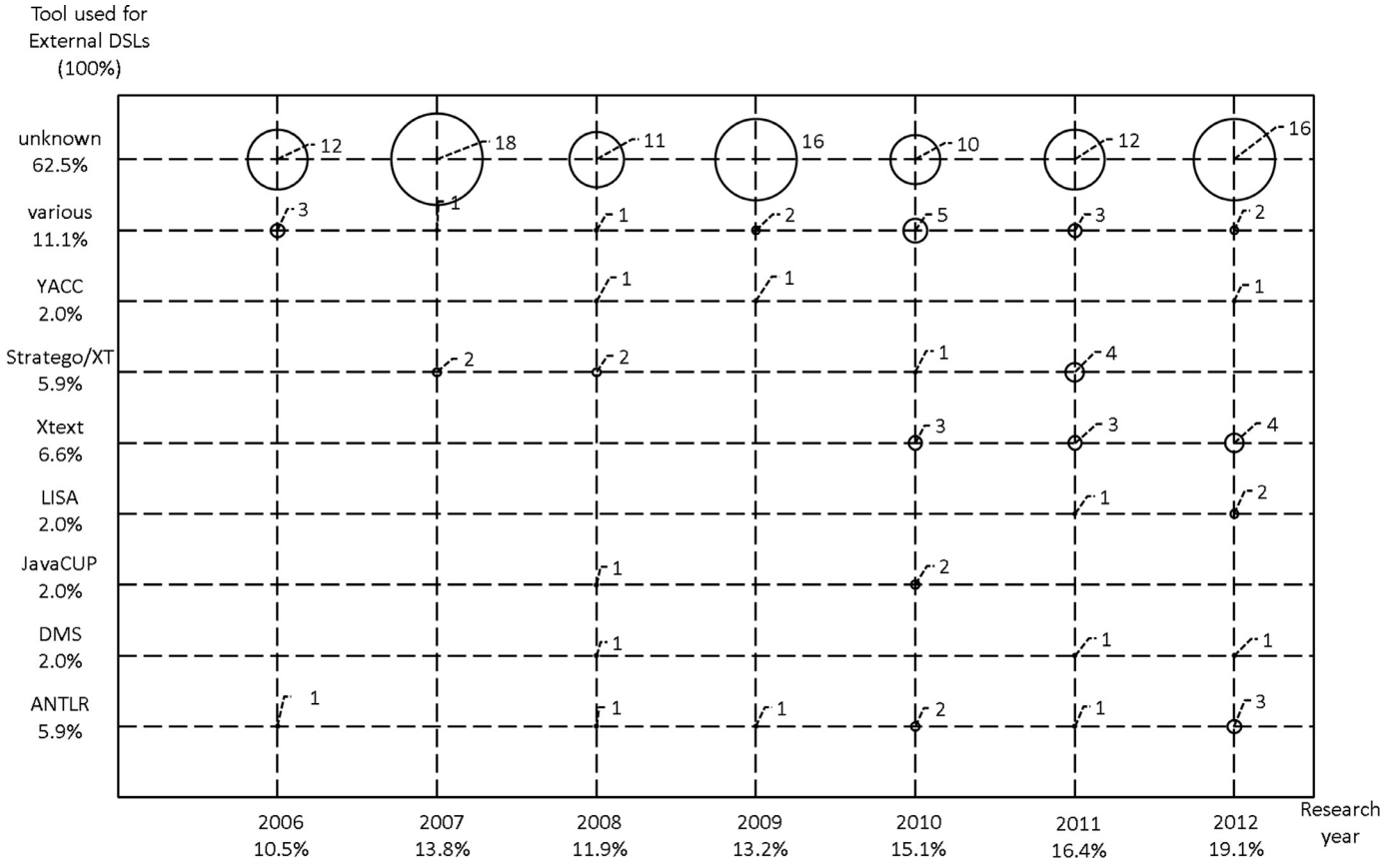

Fig. 6. Distribution of tools used for external DSL per years (RQ 1.3) , $n_5 = 152$.

Table 5
Research distribution of DSL implementation (RQ 1.3), $n_6 = 280$.

Approach	Percentage (%)
Interpreter	7.9
Compiler	28.1
Preprocessing	15.0
Embedded	34.3
Extensible compiler/interpreter	2.9
COTS	7.9
Hybrid	3.9

the external DSLs. Amongst the tools the more frequently used had been Xtext [17], ANTLR [51] and Stratego/XT [7]. This finding might suggest that DSL implementation had mostly required some specific treatment which had been unavailable in generic tools and researchers had preferred to use handwritten external DSLs. Furthermore, in [45] more than 20 DSL development tools were identified. However, most of the tools discussed in [45] except DMS [6], LISA [44,46], and Stratego/XT [7] had not been used in the primary studies included in our SMS. On the other hand, functionality between these tools (e.g., ANTLR, JavaCUP, YACC, Xtext vs. DMS, LISA, Silver, Stratego/XT) could hardly be compared due to the fact that the former tools only supported parser generation, whilst later tools could generate a whole compiler with a semantic evaluator. Nevertheless, the aforementioned tools have been used for DSL implementation and when the parser was only generated a semantic analyser had to be written from scratch.

Table 5 shows the distribution of DSL implementation patterns [45], as discussed in 280/390 ≈ 71.8% of the primary studies. Amongst the more frequently used implementation patterns have been the embedding approach (34.3%) and the compiler approach (28.1%). Other implementation approaches had been less frequently used: preprocessing (15%), COTS (7.9%), interpreter (7.9%), hybrid (3.9%), and the extensible compiler/interpreter approach (2.9%). Hence, this study doesn't support some claims that the embedding approach has prevailed over DSL implementation approaches (e.g., "*In fact, most of the common DSLs used today are designed as pure embedded programs within the structure of an existing programming language.[24]*"). In the DSL survey paper [45] 37 DSLs were discussed and classified into the aforementioned implementation patterns with somehow similar percentages: compiler approach (43%), preprocessing (20%), embedding approach (17%), interpreter (5%), COTS (5%), hybrid (5%), and extensible compiler/interpreter approach (1%).

The rest of the Figures and Tables in this section show the trends and demographics of DSL literature (RQ2). Fig. 7 shows the distributions of annual numbers of primary studies during the years 2006–2012, thus answering *RQ*2.1. The average number of primary studies per year was 55.7 with the standard deviation 18.6. Seems that the number of primary studies per year remained quite stable and showed long-lasting interest within this research field.

Table 6 shows the top cited primary studies used in our SMS using WOS (ISI Web of Science), Scopus and Google Scholar citation counts answering *RQ*2.2. In comparison, the DSL survey paper [45] has the following citations: ACM = 271, WOS = 319, Scopus = 636, Google Scholar = 1322 ($\Sigma = 2548$, AVG = 283.1). On the other hand, the DSL survey paper [45] has been cited in 91 primary studies used in this SMS (91/390 ≈ 23.3%).

Table 7 and Fig. 8 show the groups/institutions within the DSL field, measured by the number of identified primary studies in this SMS. The lines in Fig. 8 show collaborations between different

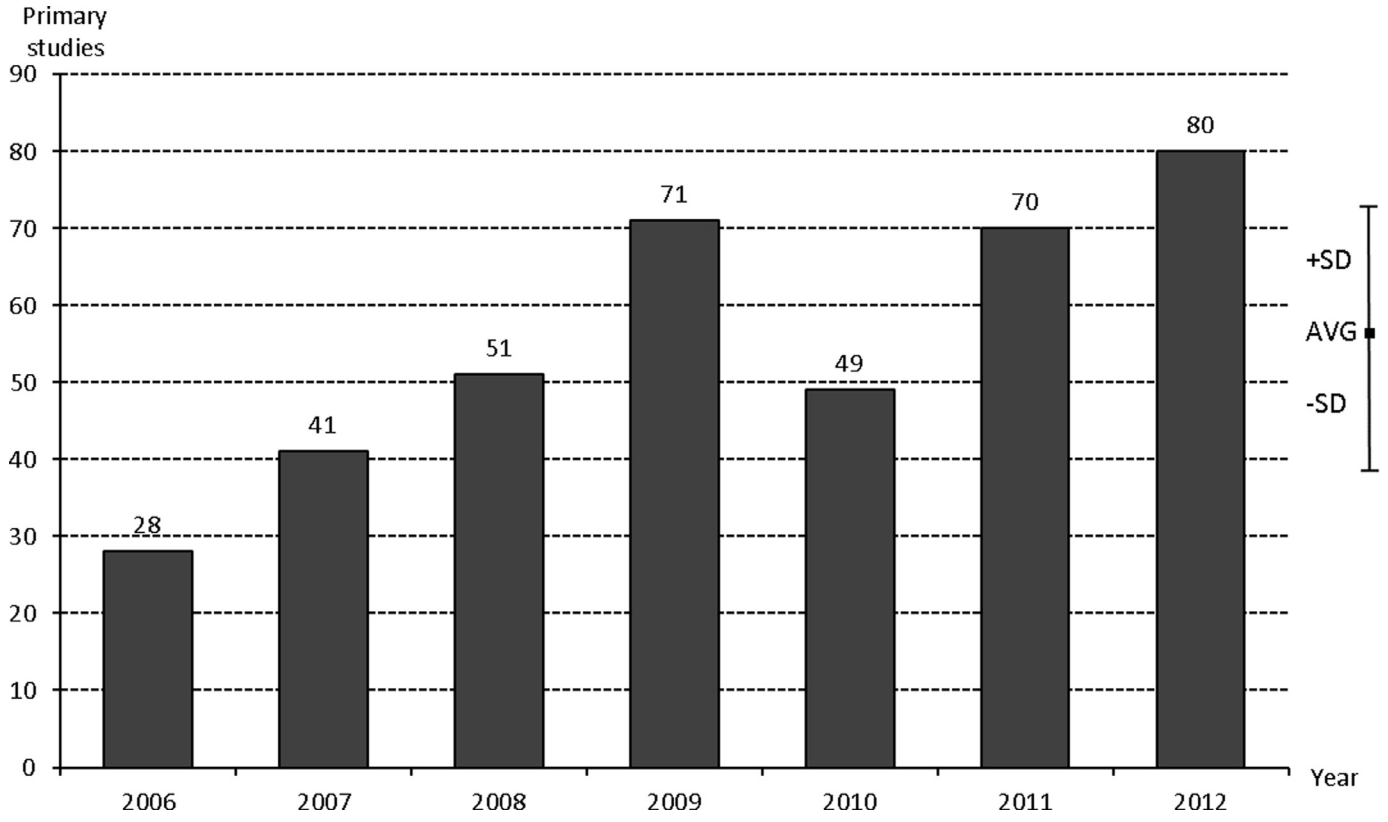

Fig. 7. Number of primary studies per years (RQ 2.1), $n = 390$.

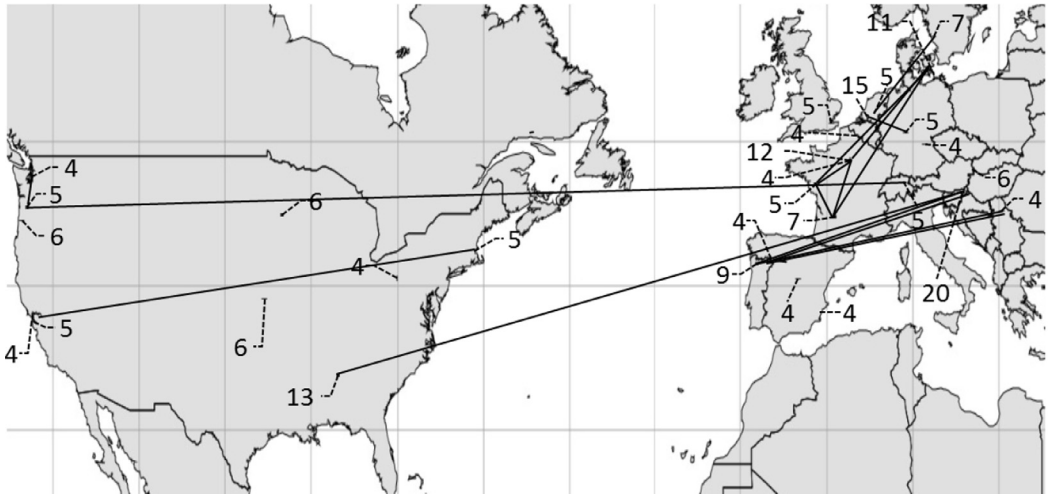

Fig. 8. Most active groups/institutions in DSL field (RQ 2.3).

researchers from different institutions on DSL research that led to joint primary studies that were included in this SMS (Table 7). Some research institutions have research centers in different cities (e.g., INRIA) or even countries (e.g., IBM Research, Microsoft Research). In this SMS we didn't differentiate between these different centers.

Fig. 9 shows the distributions of primary studies per journal over the years 2006–2012. There is a wide spectrum of different journals where DSL research had been published. Most primary studies had been published in the Computer Science and Information Systems journal (9), IEEE Transactions on Software Engineering (6), Software Practice and Experience (5), Science of Computer Programming (5), IEEE Software (5), Information and Software Technology (4), Journal of Systems and Software (4), and Journal of Supercomputing (4). Similarly, Fig. 10 shows the distributions of primary studies per conferences during the years 2006–2012. There was a wide spectrum of different conferences where DSL research had been published. Most primary studies had been published from the International Conference on Functional Programming (14), IFIP conference on Domain-Specific Languages (13), Conference on Object-Oriented Programming Systems, Languages and Applications (11), and Conference on Generative Programming: Concepts & Experiences (10). Fig. 11 shows the distributions of primary studies per workshops during the years 2006–2012. There was a wide spectrum of different workshops where DSL research had been published. Most primary studies had been published from the Workshop on Language Descriptions, Tools, and Applications (6), Workshop on Domain-Specific

Table 6
Top cited DSL primary studies by ACM, WOS, Scopus and Google scholar (RQ 2.2).

	Publication title	Year	ACM	WOS	Scopus	Google scholar	Σ	AVG per year
1	DECOR: a Method for the specification and detection of code and design smells	2010	46	66	99	243	454	90.8
2	Thespoofax language workbench: rules for declarative specification of languages and IDEs	2010	65	32	97	218	412	82.4
3	KM3: a DSL for metamodel specification	2006	76	79	86	375	616	68.4
4	Google's MapReduce programming model – revisited	2006	21	78	49	452	600	66.7
5	The pochoir stencil compiler	2011	40	32	46	138	256	64.0
6	TCS: a DSL for thespecification of textual concrete syntaxes in model engineering	2010	72		91	280	443	49.2
7	RASCAL: a domain specific language for source code analysis and manipulation	2009	27	25	73	158	283	47.2
8	SugarJ: library-based syntactic language extensibility	2011	27	17	41	70	155	38.8
9	Documenting and automating collateral evolutions in Linux device drivers	2011	49	21	19	138	227	32.4
10	An online platform for Web APIs and service mashups	2008	24	29	62	109	224	32.0
11	Comparing general-purpose and domain-specific languages: an empirical study	2010		29	55	76	160	32.0
12	An approach for the systematic development of domain-specific languages	2009	22	24	56	87	189	31.5
13	A domain-specific approach to heterogeneous parallelism	2009	31	12	18	121	182	30.3
14	Program comprehension of domain-specific and general-purpose languages: comparison using a family of experiments	2012	9	17	24	40	90	30.0
15	AmbientTalk: object-oriented event-driven programming in mobile ad hoc networks	2007	48	15	58	114	235	29.4
16	WebDSL: a case study in domain-specific language engineering	2007	28	25	28	144	225	28.1
17	Polymorphic embedding of DSLs	2008	36	10	33	93	172	24.6
18	A preliminary study on various implementation approaches of domain-specific language	2006	20	24	58	115	217	24.1
19	Attribute grammar-based language extensions for Java	2007	36	29	27	88	180	22.5
20	A domain-specific language for web APIs and services mashups	2007	24	24	20	109	177	22.1

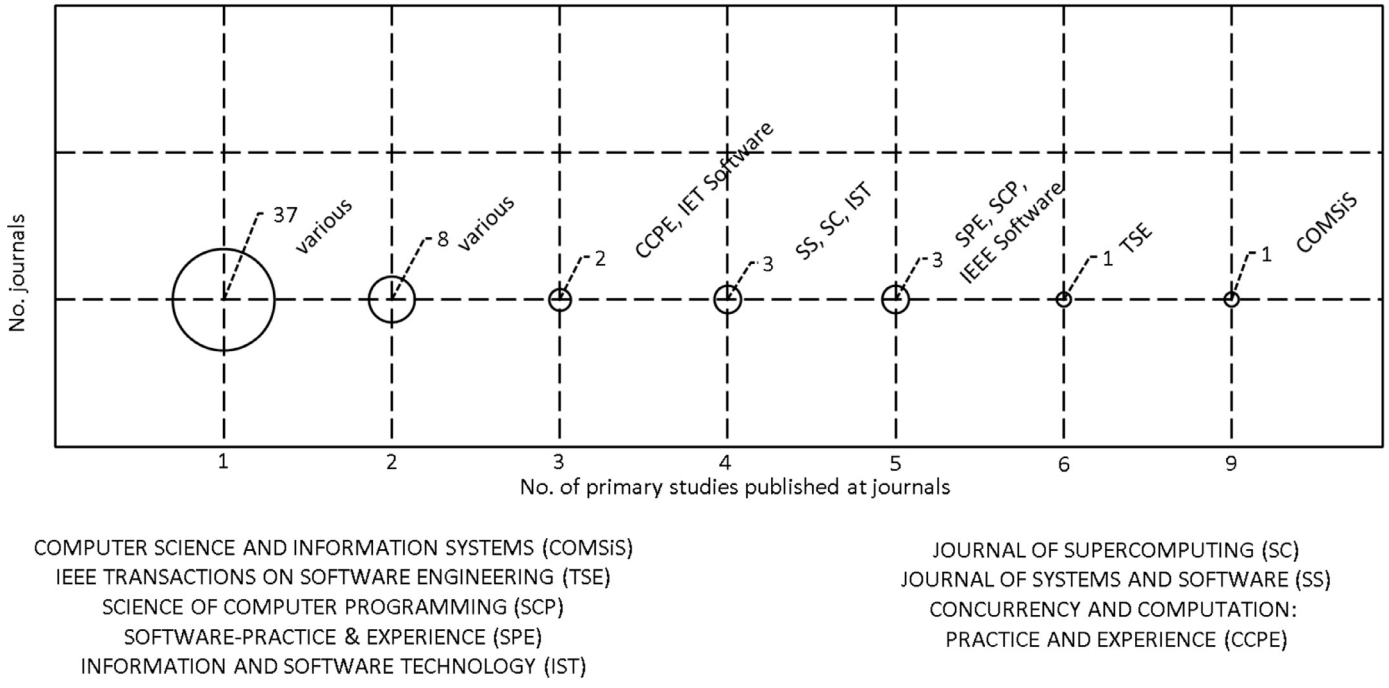

Fig. 9. Distribution of primary studies per number at journals (RQ 2.4).

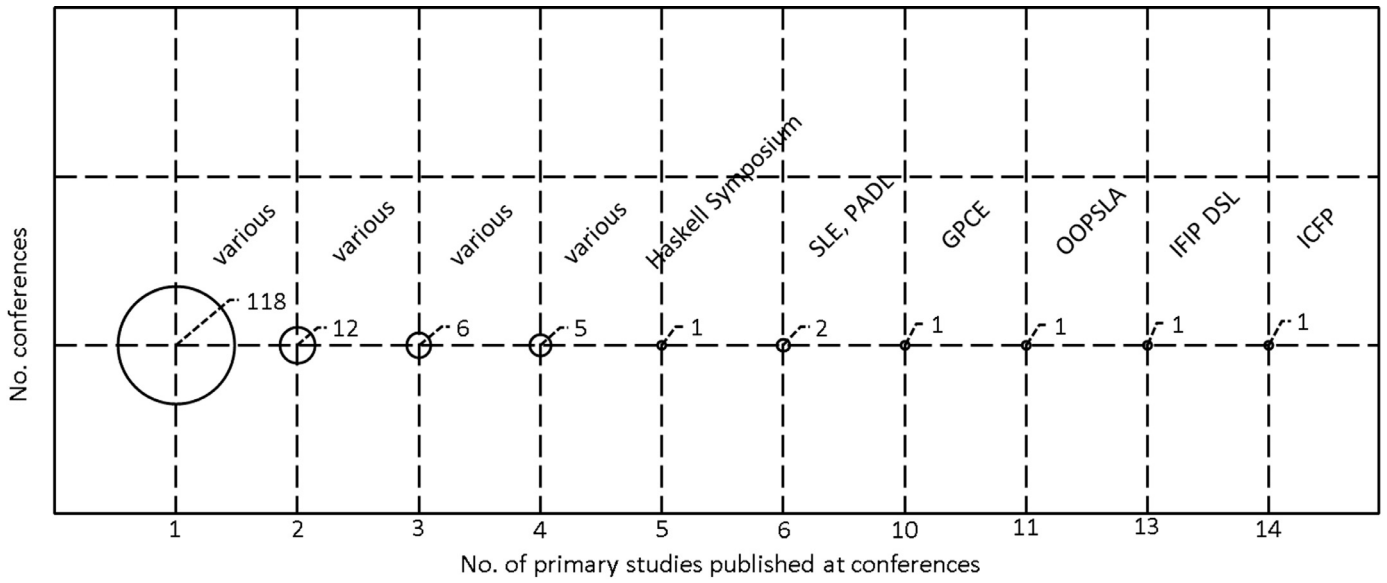

Fig. 10. Distribution of primary studies per number at conferences (RQ 2.4).

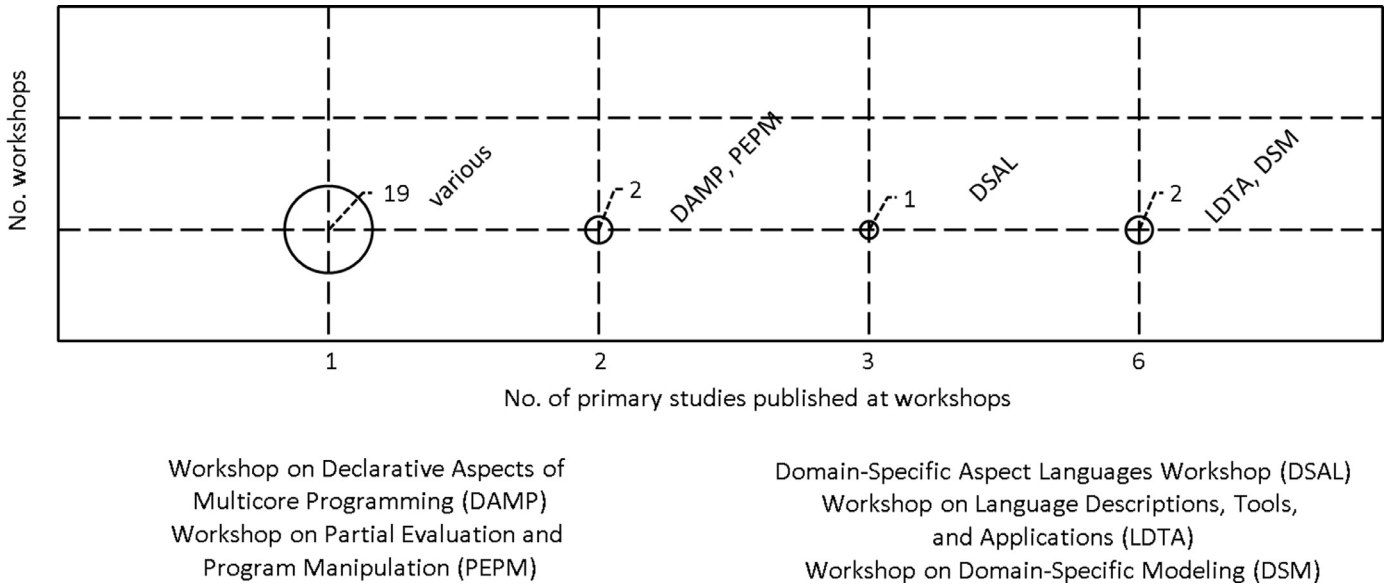

Fig. 11. Distribution of primary studies per number at workshops (RQ 2.4).

Modeling (6), and Domain-Specific Aspect languages Workshop (3). Our claim from Section 1 that DSL research has been spread over vastly different venues is clearly supported by Figs. 9,–11 (see category 'various').

5. Threats to validity

The results presented in this SMS have depended on many factors, which have been categorised into the following types of threats to the validity of the study's results: construct validity, internal validity, external validity, and conclusion validity [28,65].

5.1. Construction validity

Construct threats to validity cope with problems that might arise during research design. In the context of SMSs such threats are related to the designing of the SMS research method and especially to identifying relevant primary studies [68].

In a strict sense, our findings are valid only for our sample – DSL primary studies, which were accessible from ISI Web of Science and ACM Digital Library from the period 2006–2012. But, an important question is whether the primary studies used in our SMS are good representation of the whole population? From Figs. 7, 9–11 we can

Table 7
Institutions per number of primary studies (RQ 2.3).

Institution	Number of publications
University of Maribor	20
Delft University of Technology	15
University of Alabama at Birmingham	13
INRIA	12
University of Copenhagen	11
University of Minho	9
Chalmers University of Technology	7
University of Bordeaux	7
Oregon State University	6
University of Kansas	6
University of Minnesota	6
Vienna University of Technology	6
ETH Zurich	5
Ecole des Mines de Nantes	5
IBM Research	5
Imperial College London	5
MIT	5
National Taiwan Normal University	5
Portland State University	5
University of Marburg	5
Utrecht University	5
Carnegie Mellon University	4
Complutense University of Madrid	4
France Telecom	4
Lille University of Science and Technology	4
Microsoft Research	4
Open University of Israel	4
Polytechnic Institute of Bragança	4
Stanford University	4
University of Erlangen-Nuremberg	4
University of Murcia	4
University of Novi Sad	4

observe that our sample can be attributed as a representative sample of the whole population. In particular, Fig. 7 shows that the average number of identified primary studies per year is 55.7 with standard deviation 18.6 (hence primary studies from only one particular year do not prevail). Whilst, Figs. 9–11 show that included primary studies belonging to very diverse venues (journals, conferences, workshops). The various parts in Figs. 9–11 are always the largest part indicating that primary studies not only come from a particular journal, conference or workshop. Furthermore, by selecting a particular digital library we did not have preferences towards any research question (e.g., type of contribution, type of research, focus area). Hence, our sample can still be regarded as a good and unbiased representation of the population.

Besides this general and always presented threat there are also some more specific factors for construction validity on which researchers have greater control. For example, the search string. Due to the omissions of DSL synonyms (e.g., special purpose language, specialised language, task-specific language, application language, little language) in our search string there is a threat that some relevant primary studies were not found. However, these DSLs' synonyms have recently become unpopular (note that our research questions dealt with DSL research after 2005). In order to verify that this threat to construction validity was not too high we applied both search strings, original and expanded with synonyms, before actually conducted our SMS. With the search string expanded with synonyms the number of hits was roughly 1% bigger. Due to this reason we proceeded with the search string without DSL synonyms. Another threat to construct validity is the limited publishing period (in our case primary studies published during 2006–2012 were searched for). As our research questions addressed the DSL research after the DSL survey paper [45] was published in December 2005, the choice for 2006 is not questionable. However, we could also include primary studies which were published in the first half of the year 2013 (due to the fact that this SMS started during the Summer of 2013). However, in that case replication of our SMS could be hard in the future. We have traded the slightly bigger threat to construct validity for the much bigger possibility of replicating our study (less threat to conclusion validity). Yet another threat to construct validity was the screening of publications based on inclusion/exclusion criteria, which were defined before the study was conducted. During this process only the title, keywords and abstract were examined and there was a possibility of excluding some relevant primary study during this process. In order to mitigate this threat, whenever we were unsure whether a publication should be excluded or not we opted for temporary inclusion. However, during classification when a whole publication was read we may still have excluded it.

5.2. Internal validity

Internal threats to validity cope with problems that might arise during data extraction. Extracting data from primary studies might be problematic, especially if the qualities of primary studies are low or the requested information is buried within a paper. One such very broad data extraction was done during the screening phase and information as to whether a publication was about DSL research or not, was extracted from the title, keywords, and abstract. More fine-grained data extraction was performed during the classifications of selected primary studies. This phase is one of the more error-prone phases when performing SMSs due to the lack of standard terminology, missing information, or badly presented primary studies. In order to minimise the internal threats to validity, the extracted data from the whole primary study were classified by the DSL expert (see discussion in Section 3.2). Although this might inherently introduce bias we felt that in a such manner classification would be more consistent. Furthermore, criteria for classification on the type of research were defined in more detail than in many other SMSs (see discussion in Section 3.1 about two level classification) to mitigate this threat to validity. Moreover, as the classification of primary studies is a time-intensive process where fatigue may lead to misclassification, an additional constraint was placed within the research method (classification time was restricted to two consecutive hours followed by one hour breaks) in order to mitigate this threat to validity. On the other hand, analysis of the classified primary studies requires only the use of descriptive statistics. Hence, it is less likely that we made some errors during this step.

5.3. External validity

External threats to validity cope with problems that might arise during conclusion generalisation. The conclusions drawn from this study are DSL research-specific and cannot be generalised within other research topics. This is even true, for closely related fields of DSMLs and visual/graphical languages. As in our SMS we excluded modeling languages, as well as graphical languages, our findings should not be generalised to these closely related research fields. Although some conclusions could be generalised to a broader topic of SE (e.g., lack of evaluation research studies) we didn't do such general conclusions.

5.4. Conclusion validity

Conclusion threats to validity (another term is reliability) cope with problems that might arise when deriving conclusions and whether the SMS can be repeated. Some factors that can lead to incorrect conclusions, either by identifing wrong relationships, or by missing existing relationships are related to the research method and incorrect data extraction. Both were discussed under construction and internal validities. Furthermore, by a two-level classification of primary studies (on the first level: empirical and non-empirical study),

we have alleviated to some extent the conclusion validity as wrong or missing relationships are less likely to be identified. Another factor which helps us to mitigate this threat is that the number of classified primary studies was much bigger than in similar SMSs. An absent or wrongly classified study would not skew statistics as much as in the case of a small number of classified primary studies (e.g., 13 in [5]). Hence, the chances for wrong conclusions are smaller. Replication of this study can be achieved as our research method was detailed (the protocol is available at [39]) and we didn't omit any important information. The raw data are available at [59].

6. Conclusions and future work

In this work SMS on DSLs was presented with the aim of providing the DSL research community with an unbiased, objective and systematic overview of DSL research done during period 2006–2012. We strongly believe that within each research topic such SMSs should be periodically performed to make researchers aware of the amount of work done, the progress, and to find out possible gaps in the research. The main findings of our SMS on DSLs are:

- Research about DSL integration with other SE process is lacking, as well as measuring the effectiveness of DSL approaches.
- Clear lack of evaluation research, in particular controlled experiments.
- Amongst different DSL development phases the following phases have been insufficiently investigated: domain analysis, validation and maintenance.
- Lack of use regarding formal methods within domain analysis and in the semantic description of DSLs.

We encourage the DSL researchers to start addressing the identified gaps to enable practitioners to understand the effectiveness and efficiencies of DSLs.

Last, but not least, we suggest some improvements in performing SMSs and enhancing their reliability. In particular, we suggest that the keywording process would be better if replaced by taking into account survey papers within a research field and/or by contacting experts within that research field. Furthermore, we have enhanced the reliability of SMSs by coarse-grained classification on empirical or non-empirical research. Finally, we encourage those interested in performing SRs for building appropriate tools for better supporting the whole process.

As future work we would like to perform SMS on DSMLs where various differences between DSLs and DSMLs can be identified (e.g., amount of research work done, different emphases on types of contributions, different emphases on the types of research, differences regarding research topics (focus area), maturity of the tools).

References

[1] A. Ampatzoglou, S. Charalampidou, I. Stamelos, Research state of the art on GoF design patterns: a mapping study, J. Syst. Softw. 86 (7) (2013) 1945–1964.
[2] A. Barišić, V. Amaral, M.G. ao, Usability evaluation of domain-specific languages, in: Proceedings of Simpósio de Estudantes de Doutoramento em Engenharia de Software (SEDES 2012 @ QUATIC 2012), 2012, pp. 342–347.
[3] A. Barišić, V. Amaral, M.G. ao, B. Barroca, Evaluating the usability of domain-specific languages, in: M. Mernik (Ed.), Formal and Practical Aspects of Domain-Specific Languages: Recent Developments (Chapter 14), IGI Global, 2013, pp. 386–407.
[4] S. Barney, K. Petersen, M. Svahnberg, A. Aurum, H. Barney, Software quality trade-offs: a systematic map, Inf. Softw. Technol. 54 (7) (2012) 651–662.
[5] E. Barreiros, A. Almeida, J. Saraiva, S. Soares, A systematic mapping studies on software engineering testbeds, in: Proceedings of the International Symposium on Empirical Software Engineering and Measurement (ESEM'11), 2011, pp. 107–116.
[6] I.D. Baxter, C. Pidgeon, M. Mehlich, DMS: program transformation for practical scalable software evolution, in: Proceedings of the 26th International Conference on Software Engineering (ICSEÆ04), 2004, pp. 625–634.
[7] M. Bravenboer, K.T. Kalleberg, R. Vermaas, E. Visser, Stratego/XT 0.17. a language and toolset for program transformation, Sci. Comput. Program. 72 (1–2) (2008) 52–70.
[8] P. Brereton, B. Kitchenham, D. Budgen, M. Turner, M. Khalil, Lessons from applying the systematic literature review process within the software engineering domain, J. Syst. Softw. 80 (4) (2007) 571–583.
[9] B.R. Bryant, J. Gray, M. Mernik, P.J. Clarke, R.B. France, G. Karsai, Challenges and directions in formalizing the semantics of modeling languages, Comput. Sci. Inf. Syst. 8 (2) (2011) 225–253.
[10] D. Budgen, M. Turner, P. Brereton, B. Kitchenham, Using mapping studies in software engineering, in: Proceedings of the 20th Annual Meeting of the Psychology of Programming Interest Group (PPIG'08), 2008, pp. 195–204.
[11] J.L.C. Izquierdo, J. Cabot, J.J. López-Fernández, J.S. Cuadrado, E. Guerra, J. de Lara, Engaging end-users in the collaborative development of domain-specific modelling languages, in: Proceedings of International Conference on Cooperative Design, Visualization and Engineering (CDVE'13), 2013, pp. 101–110.
[12] J.C. Carver, E. Syriani, J. Gray, Assessing the frequency of empirical evaluation in software modeling research, in: Proceedings of the First Workshop on Experiences and Empirical Studies in Software Modelling, 2011, p. 5.
[13] C. Catal, D. Mishra, Test case prioritization: a systematic mapping study, Softw. Qual. J. 21 (3) (2013) 445–478.
[14] K. Czarnecki, U. Eisenecker, Generative Programming: Methods, Tools and Applications, Addison-Wesley, 2000.
[15] I. Čeh, M. Črepinšek, T. Kosar, M. Mernik, Ontology driven development of domain-specific languages, Comput. Sci. Inf. Syst. 8 (2) (2011) 317–344.
[16] A. van Deursen, P. Klint, J. Visser, Domain-specific languages: an annotated bibliography, ACM SIGPLAN Not. 35 (6) (2000) 26–36.
[17] S. Efftinge, M. Völter, oAW xText: a framework for textual DSLs, in: Proceedings of the Workshop on Modeling Symposium at Eclipse Summit, 2006.
[18] E. Engström, P. Runeson, Software product line testing û a systematic mapping study, Inf. Softw. Technol. 53 (1) (2011) 2–13.
[19] S. Erdweg, T. van der Storm, M. Völter, L. Tratt, R. Bosman, W.R. Cook, A. Gerritsen, A. Hulshout, S. Kelly, A. Loh, G. Konat, P.J. Molina, M. Palatnik, R. Pohjonen, E. Schindler, K. Schindler, R. Solmi, V. Vergu, E. Visser, K. van der Vlist, G. Wachsmuth, J. van der Woning, Evaluating and comparing language workbenches: existing results and benchmarks for the future, Comput. Lang. Syst. Struct. 44 (Part A) (2015) 24–47.
[20] A.M. Fernández-Sáez, M. Genero, M.R.V. Chaudron, Empirical studies concerning the maintenance of UML diagrams and their use in the maintenance of code: a systematic mapping study, Inf. Softw. Technol. 55 (7) (2013) 1119–1142.
[21] M. Fowler, Domain Specific Languages, Addison-Wesley, 2010.
[22] P. Gabriel, M.G. ao, V. Amaral, Do software languages engineers evaluate their languages? in: Proceedings of XIII Congreso Iberoamericano en "Software Engineering" (ClbSE 2010), 2010, pp. 149–162.
[23] V. Garousi, A. Mesbah, A. Betin-Can, S. Mirshokraie, A systematic mapping study on web application testing, Inf. Softw. Technol. 55 (8) (2013) 1374–1396.
[24] D. Ghosh, DSL for the uninitiated, Commun. ACM 54 (7) (2011) 44–50.
[25] J. Gray, J.-P. Tolvanen, S. Kelly, A. Gokhale, S. Neema, J. Sprinkle, Domain-Specific Modeling. In Handbook of Dynamic System Modeling, CRC Press, 2007.
[26] J. Greenfield, K. Short, Software Factories: Assembling Applications with Patterns, Models, Frameworks, and Tools, Wiley Publishing, 2004.
[27] P. Hudak, Building domain-specific embedded languages, ACM Comput. Surv 28 (4es) (1996) Article No. 196.
[28] A. Jedlitschka, M. Ciolkowski, D. Pfahl, Reporting experiments in software engineering, in: F. Shull, J. Singer, D. Sjøberg (Eds.), Guide to Advanced Empirical Software Engineering, chapter 8, Springer-Verlag, London, 2008.
[29] K.C. Kang, S.G. Cohen, J.A. Hess, W.E. Novak, A.S. Peterson, Feature-Oriented Domain Analysis (FODA) Feasibility Study, Techical report cmu/sei-90-tr-21, Software Engineering Institute, Carnegie Mellon University, 1990.
[30] V. Karakoidas, D. Mitropoulos, P. Louridas, D. Spinellis, A type-safe embedding of SQL into Java using the extensible compiler framework J%, Comput. Lang. Syst. Struct. 41 (2015) 1–20.
[31] B. Kitchenham, S. Charters, Guidelines for performing systematic literature reviews in software engineering, Keele University, 2007.
[32] B. Kitchenham, P. Brereton, D. Budgen, M. Turner, J. Bailey, S. Linkman, Systematic literature review in software engnering – a systematic literature review, Inf. Softw. Technol. 51 (1) (2009) 638–651.
[33] B. Kitchenham, R. Pretorius, D. Budgen, P. Brereton, M. Turner, M. Niazi, S. Linkman, Systematic literature reviews in software engineering – a tertiary study, Inf. Softw. Technol. 52 (8) (2010) 792–805.
[34] B. Kitchenham, D. Budgen, P. Brereton, Using mapping studies as the basis for further research – a participant-observer case study, Inf. Softw. Technol. 53 (6) (2011) 638–651.
[35] A. Kleppe, Software Language Engineering: Creating Domain-Specific Languages Using Metamodels, Addison-Wesley Professional, 2008.
[36] T. Kosar, P.E.M. López, P.A. Barrientos, M. Mernik, A preliminary study on various implementation approaches of domain-specific language, Inf. Softw. Technol. 50 (5) (2008) 390–405.
[37] T. Kosar, N. Oliveira, M. Mernik, M.J.V. Pereira, M. Črepinšek, D. da Cruz, P.R. Henriques, Comparing general-purpose and domain-specific languages: an empirical study, Comput. Sci. Inf. Syst. 7 (2) (2010) 247–264.
[38] T. Kosar, M. Mernik, J.C. Carver, Program comprehension of domain-specific and general-purpose languages: comparison using a family of experiments, Empir. Softw. Eng. 17 (3) (2012) 276–304.
[39] T. Kosar, S. Bohra, M. Mernik, Protocol of Systematic Mapping Study on DSLs, Available at http://lpm.feri.um.si/projects/DSL_SMS_Protocol.pdf.
[40] M.A. Laguna, Y. Crespo, A systematic mapping study on software product line evolution: from legacy system reengineering to product line refactoring, Sci. Comput. Program. 78 (8) (2013) 1010–1034.

[41] L.B. Lisboa, V.C. Garcia, D. Lucredio, E.S. de Almeida, S.R. de Lemos Meira, R.P. de Mattos Fortes, A systematic review of domain analysis tools, Inf. Softw. Technol. 52 (1) (2010) 1–13.
[42] A.P. van der Meer, Domain Specific Languages and their Type Systems (Ph.d. thesis), Eindhoven University of Technology, 2014.
[43] A. Mehmood, D.N.A. Jawawi, Aspect-oriented model-driven code generation: a systematic mapping study, Inf. Softw. Technol. 55 (2) (2013) 395–411.
[44] M. Mernik, M. Lenič, E. Avdičaušević, V. Žumer, LISA: an interactive environment for programming language development, in: Proceedings of 11th International Conference on Compiler Construction, (CC'02), 2002, pp. 1–4.
[45] M. Mernik, J. Heering, A.M. Sloane, When and how to develop domain-specific languages, ACM Comput. Surv. 37 (4) (2005) 316–344.
[46] M. Mernik, V. Žumer, Incremental programming language development, Comput. Lang. Syst. Struct. 31 (1) (2005) 1–16.
[47] M. Mernik, Formal and Practical Aspects of Domain-Specific Languages: Recent Developments, IGI Global, 2013.
[48] P.A. da Mota Silveira Neto, I. do Carmo Machado, J.D. McGregor, E.S. de Almeida, S.R. de Lemos Meira, A systematic mapping study of software productlines testing, Inf. Softw. Technol. 53 (5) (2011) 407–423.
[49] L.M. do Nascimento, D.L. Viana, P.A.M. Silveira Neto, D.A.O. Martins, V.C. Garcia, S.R.L. Meira, A systematic mapping study on domain-specific languages, in: Proceedings of the 7th International Conference on Software Engineering Advances (ICSEA'12), 2012, pp. 179–187.
[50] R.F. Paige, D.S. Kolovos, F.A.C. Polack, A tutorial on metamodelling for grammar researchers, Sci. Comput. Program. 96 (4) (2014) 396–û416.
[51] T. Parr, Language Implementation Patterns: Create Your Own Domain-Specific and General Programming Languages, The Pragmatic Bookshelf, 2010.
[52] K. Petersen, R. Feldt, S. Mujtaba, M. Mattsson, Systematic mapping studies in software engineering, in: Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering (EASE'08), 2008, pp. 71–80.
[53] K. Petersen, S. Vakkalanka, L. Kuzniarz, Guidelines for conducting systematic mapping studies in software engineering: an update, Inf. Softw. Technol. 64 (2015) 1–18.
[54] H. Prähofer, R. Schatz, C. Wirth, D. Hurnaus, H. Mössenböck, Monaco – a domain-specific language solution for reactive process control programming with hierarchical components, Comput. Lang. Syst. Struct. 39 (3) (2013) 67–94.
[55] R. Pretorius, D. Budgen, A mapping study on empirical evidence related to the models and forms used in the UML, in: Proceedings of the International Symposium on Empiricial Software Engineering and Measurement (ESEM'08), 2008, pp. 342–344.
[56] A.R. da Silva, Model-driven engineering: a survey supported by a unified conceptual model, Comput. Lang. Syst. Struct. 43 (2015) 139–155.
[57] D.C. Schmidt, Model-driven engineering, IEEE Comput. 39 (2) (2006) 25–31.
[58] J. Sprinkle, M. Mernik, J.-P. Tolvanen, D. Spinellis, What kinds of nails need a domain-specific hammer? IEEE Softw. 26 (4) (2009) 15–18.
[59] Raw data of Systematic Mapping Study on DSLs. Available at http://lpm.feri.um.si/projects/DSL_SMS_Protocol.pdf.
[60] R. Tairas, J. Cabot, Corpus-based analysis of domain-specific languages, Softw. Syst. Model. 24 (2) (2015) 889–904.
[61] E. Vacchi, W. Cazzola, Neverlang: a framework for feature-oriented language development, Comput. Lang. Syst. Struct. 43 (2015) 1–40.
[62] M.P. Ward, Language-oriented programming, Softw. – Concepts Tools 15 (1995) 147–161.
[63] D. Weiss, C.T.R. Lay, Software Product Line Engineering, Addison-Wesley, 1999.
[64] R. Wieringa, N. Maiden, N. Mead, C. Rolland, Requirements engineering paper classification and evaluation criteria: a proposal and a discussion, Requir. Eng. 11 (1) (2006) 102–107.
[65] C. Wohlin, P. Runeson, M. Höst, M.C. Ohlsson, B. Regnell, A. Wesslén, Experimentation in Software Engineering: An Introduction, Kluwer Academic Publishers, Norwell, MA, USA, 2000.
[66] C. Wohlin, P. Runeson, P.A. da Mota Silveira Neto, E. Engstrom, I. do Carmo Machado, E.S. de Almeida, On the reliability of mapping studies in software engineering, J. Syst. Softw. 86 (10) (2013) 2594–2610.
[67] S. Yoo, M. Harman, Regression testing minimization, selection and prioritization: a survey, Softw. Test. Verif. Reliab. 22 (2) (2012) 67û–120.
[68] H. Zhang, M.A. Babar, P. Tell, Identifying relevant studies in software engineering, Inf. Softw. Technol. 53 (6) (2011) 625–637.