PROTOCOL FOR A SYSTEMATIC LITERATURE REVIEW ON

Execution of UML Models

VERSION 0.5

FEDERICO CICCOZZI (MÄLARDALEN UNIVERSITY)
IVANO MALAVOLTA (GRAN SASSO SCIENCE INSTITUTE)
BRAN SELIC (MALINA SOFTWARE CORP.)

PROTOCOL FOR A SYSTEMATIC LITERATURE REVIEW ON

Execution of UML Models

VERSION 0.5, FEBRUARY 29, 2016

ABSTRACT

This document describes the review protocol of a systematic literature review (SLR) on the execution of UML models.

KEYWORDS

Systematic Literature Review, SLR, UML, Alf, fUML, execution, code generation, compilation.

DOCUMENT VERSION CONTROL

Document status	**Version #**	**Date**	**Changes from previous version**
Draft	0.5	February 26, 2015	None
Draft	0.5	March 2, 2015	Separation of sections in files
First release	0.5	March 18, 2015	Concluded first complete version of the protocol
Second release	0.5	June 23, 2015	Addressed reviewers comments and other modifications to search & selection strategy
Third release	0.5	December 4, 2015	Alignment to data extraction process

Contents

List of Figures

List of Tables

1 Background and rationale

Software is everywhere and characterizes our everyday life in many ways. Several different methodologies have been defined and used for the development of software with the common goal of providing means to eventually make it executable on a designated platform. In the early 50's executables in terms of machine languages were written by hand. The need for simplification by abstraction led to the creation of a plethora of programming languages, many of which are still in use. These languages used higher-level abstractions to represent the actual code used by the underlying computers. We went from first-generation languages, represented by machine code languages, to second-generation languages, represented by assembly languages, to eventually end up in third-generation languages, known as high-level programming languages. Code written using second- and third-generation programming languages was not directly executable, but needed to be brought to a format that the machine would understand. In fact, when it comes to the execution of a software artefact, no matter which higher-level programming language is exploited, there are usually two ways to go:

- **Interpretative execution:** the software artefact is interpreted to be directly executed on the target platform;
- **Compilative execution:** the software artefact is first compiled in order to express it as a set of low level instructions that the machine would understand and which are then executed.

Starting from the late 90's it became clear that the very fast growth of software's complexity would sooner or later lead to the need of defining ways to develop more powerful and more flexible approaches. Once again the need of further abstraction arose.

In this direction Model-Driven Engineering (MDE) started to get a foothold as a promising way to tackle the inability of third-generation languages to effectively mitigate complexity and express domain-specific concepts [2]. MDE aims at easing software development by promoting models as primary artifacts in the development. Rules and constraints for building models have to be properly described through a corresponding language definition. To this end, a metamodel is often used to describe the set of available concepts and well-formedness rules to which a correct model must conform [2]. By dealing directly with domain-specific abstractions, models can reduce complexity and allow the developer to focus on the concepts that matter in the design of the application [2].

There is a rather heated ongoing debate in the software community about whether or not there is a difference between programming languages and modeling languages, particularly when dealing with executable modeling languages. To understand the difference between the two types of languages, it is useful to first examine the primary purposes of both models and programs.

For programs, it is quite clear: the primary purpose of a program is to unambiguously instruct a computer on what needs to be done. In fact, this is why programming is often referred to as coding; that is, translating a design specification into a (hopefully) equivalent form expressed using the language of a computer.

Models, on the other hand, serve a richer set of purposes:

- Models support understanding. Because they are selective abstractions of reality, models are often used to support the analysis of both problems and solutions. Note that, since different stakeholders may be interested in different aspects of a system, there may be multiple different models of the same system, each emphasizing the aspects that are of interest to a particular group of stakeholders.
- Models support communication between stakeholders. To better support analysis, models must be readily understandable by their human audience. Consequently, the concrete syntax of useful modeling languages tends to be much closer to human cognition than the syntax of programming languages. This is the reason why many modeling languages utilize a graphical syntax (at least in part), since graphical renderings often convey information more succinctly and more directly to human readers compared to text, especially when dealing with

structural modeling. Hence, well-designed modeling languages place much more emphasis on understandability than programming languages.

- Models can be used to predict interesting properties of systems. Models are often used to make predictions about the interesting properties of a system or a design. Because they abstract away detail that is irrelevant to the concern on hand, it is possible to make predictions long before committing time and resources to the construction of the corresponding real system.
- Models can be used as design specifications. This purpose of models differs in a significant way from the ones listed above: a model serving as design specification must be detailed and accurate enough to be unambiguous. Needless to say, the more detail it contains, the more difficult it is to understand a model. Fortunately, one of the key features of computer-based models of software systems is that, if they are specified in an executable modeling language, they can gradually evolve from early very abstract models into specifications that are precise and detailed enough to be directly translated into executable code. This progression from abstract model to the actual system can be seamless; that is, the model gradually morphs into the system it was modeling, avoiding along the way those error-prone discontinuities in skills, tools, or media, which are the bane of more traditional engineering technologies. However, this capability presents a very unique design challenge for modeling languages: it must be possible to create both very abstract and possibly incomplete models as well as very detailed and precise ones.

Based on the above, we can perhaps summarize the principal difference between modeling and programming languages by saying that well-designed modeling languages tend to be much more human centric than programming languages.

One of the main goals of the MDE paradigm is to promote automation, typically achieved through model manipulations, in the development process. An example of this could be the automatic generation of executable artefacts from design models by means transformations between models. A model transformation translates a source model to a target model while preserving their well-formedness [3].

By shifting the focus of the development from hand-written code to models, a natural consequence would have been the definition of similar execution mechanisms, namely *interpretative* and *compilative*, for modelling languages as well. Instead, due to several reasons, such as reuse of existing compilers/interpreters/virtual machines, a third mechanism, namely **translational execution**, was introduced and historically preferred in MDE processes. For translational execution it is meant the translation of a modelling artefact to code usually conforming to a third-generation programming language, different from the modelling language itself, and thereby its execution on the target machine after interpretation or compilation.

When it comes to modelling means, in the myriad of general-purpose and domain-specific modelling languages, the Unified Modeling Language (UML)[1], standardized by the Object Management Group (OMG), has emerged and established itself as de-jure standard in industrial development of software systems [4]. This was partially thanks to its versatility, which enables (i) its usage as general-purpose language, and (ii) the possibility to customize it through the so-called profiling mechanisms [5] to give it domain specificity. Although helping in raising the level of abstraction, initially UML was not sufficiently powerful to produce executable programs as traditional third-generation programming languages were and therefore it had to rely on them to achieve executable artefacts, usually through semi-automatic generation of code (i.e., translational execution) making intrepretive and compilative execution hard to achieve. The introduction of UML2 together with the definition of (i) a formal specification of the execution semantics for a subset of UML2 through the Foundational Subset For Executable UML Models (fUML)[2] and (ii) a textual action language, the Action Language for Foundational UML (ALF)[3], to fully express in a compact manner complex

[1] http://www.uml.org/
[2] http://www.omg.org/spec/FUML/
[3] http://www.omg.org/spec/ALF/

execution behaviors and their algorithmic part, provided the needed tools for achieving models that are fully executable even through intepretive and compilative mechanisms.

In this Systematic Literature Review (SLR) we aim at identifying, classifying, and evaluating existing methods and techniques for the execution of models defined through the UML family of languages (i.e., UML2, UML2 profiles, fUML, ALF) as deployable running artefacts.

1.1 Existing systematic literature reviews on the topic

Even if there are systematic reviews and surveys on UML-based methods and techniques, each of them with different perspectives and goals, we could not identify any systematic review on the execution of UML models as deployable running artefacts. To the best of our knowledge, to date there are no extended surveys that explicitly focus on this topic. In the following we provide an overview of the research works that are related to ours either in terms of object of study (i.e., UML models) or methodology (i.e., systematic literature review).

In [6] the authors present a systematic literature review about approaches for UML models consistency management. This research has been carried out by following the guidelines proposed by Kitchenham et al. [7] and it explores 44 primary studies. Among the main results of this work it emerged that the line of research for UML models consistency is highly active and promising (in 2009), and some limitations and future research directions have been pointed out. Based on the identified limitations, the authors of this study (i) proposed a set of recommendations for future UML models consistency management and (ii) presented a preliminary formal approach for the handling of inconsistency problems via model transformations and rewriting logic.

Budgen et al. carried out a combined study in which both a systematic mapping and a systematic literature review were performed to identify relevant empirical studies about UML [8]. Firstly, the authors mapped the existing literature with the main goal of investigating the extent to which the notations of the UML and their usefulness have been empirically studied. Secondly, the authors conducted a systematic literature review based on the categories identified in the previous step. The systematic literature review covered empirical studies published before the end of 2008. They analysed 49 primary studies in terms of the categories of the mapping for which the authors had enough articles (i.e., metrics, comprehension, model quality, methods and tools and adoption). The main result of this study is that "researchers tend to use the UML as given and seem reluctant to ask questions that might help to make it more effective".

The quality of UML models have been investigated in a systematic literature review proposed by Genero et al. [9]. Also this work follows the guidelines proposed by Kitchenham et al. [7] and it uses six different electronic databases as data sources. The study search and selection process resulted in 266 selected primary studies; those studies have been classified according to the following categories: type of quality, context of study, type of UML diagram, type of research result, research method and research goal. The main results of this study are that (i) most research focuses on semantic quality, rather than on semantic completeness and (ii) more empirical research is needed to develop a theoretical understanding of conceptual model quality.

On a different line, a Web survey targeted at analysts familiar with OO techniques and UML in particular has been conducted in [10]. The main goal of this study is to understand how and why UML analysis components are used by practitioners. The analysis of obtained data resulted in an assessment of the usage levels of the various diagram of UML (the highest usage levels are for use case diagrams and class diagrams, the lowest for collaboration diagrams). Also, contrary to claims in many other articles in the literature, developers seem to believe that UML diagrams can be understood by clients. Finally, it emerged that the complexity of UML is a concern that may be solved by introducing more programs to help practitioners and their clients to learn UML and how to use it more effectively.

A large scale survey has been carried by Gorschek et al. on the use of software design models in software development practice [11]. The survey includes also UML as a means for creating design models. The survey was executed through the creation of an on-line questionnaire that has been

advertised through a variety of means including YouTube and Twitter. Potential subjects of the survey were software developers with experience in software development and in using any object-oriented programming language. The authors of this work collected the answers provided by 3785 developers. The main conclusion of this work is that about 50% and about 70% of respondents never or rarely use design models before coding, respectively. This result reveals that the use of models in general, and the UML in particular, does not seem to be standard practice and de facto standard in software development. By starting from the same 3875 provided answers, in [12] the authors focused on how developers understand, use, and apply object-oriented concepts (e.g., encapsulation, class size as measured by number of methods, and depth of classes in the inheritance hierarchy). Among the various results, this study provides evidence about the fact that developers generally follow the advice on the importance of hiding representation. Also, half of the participants agree with the importance of observing, monitoring and limiting class depth and size,whereas others do not consider it.

A survey of existing consistency checking techniques between UML models is proposed in [13]. Identified categories were mainly related to consistency types (e.g., inter-model consistency, intra-model consistency, semantic consistency, etc.) and analysis parameters (e.g., static or dynamic, UML version, intermediate representation, etc.). The results of the classification reveal that most of the techniques focus on inter- and intra-level consistency validation between UML models; also, the majority of UML consistency checking techniques are based on a tool that applies a continuous monitoring strategy and helps in quick validation.

In [14] a survey of the existing literature about the transformation of UML models into XML schemas is presented. Selected articles have been compared mainly on the basis of Czarnecki's feature model for representing the various design choices for a model transformation approach [15]. Basically, the main contribution of this paper is to present an extensive overview and comparison of existing transformations from UML models to XML schemas. One result of this research is that, according to the selected papers, all approaches proposing a transformation of UML models to XML schemas consider UML class diagrams only. Other UML diagrams are considered as well, but they are translated into XML documents but not into XML schemas.

1.2 The need for an SLR on the execution of UML models

Without knowing the potential of execution of UML models as deployable running artefacts (especially from a practical and usability perspective), the transfer of solutions to practice may be severely hampered. Hence, a systematic literature review about existing research and tools can be an invaluable instrument for assessing existing methods and techniques for execution of UML models.

With this systematic literature review we aim to identify, classify, and understand existing research on the execution of UML models. Those activities will help researchers and practitioners in identifying limitations and gaps of current research [7] on the execution of UML models, its potential, and its applicability in practice in the context of real-world projects.

2 Research implementation

In this section we describe method and settings to carry out this research work.

2.1 Process

This research will be carried out by following the process shown in Figure 1; it can be divided into three main phases, which are well-established when it comes to systematic literature reviews [16, 17]: planning, conducting, and documenting.

Each phase has a number of output artefacts, e.g., the planning phase produces the review protocol described in this document. In order to mitigate potential threats to validity and possible

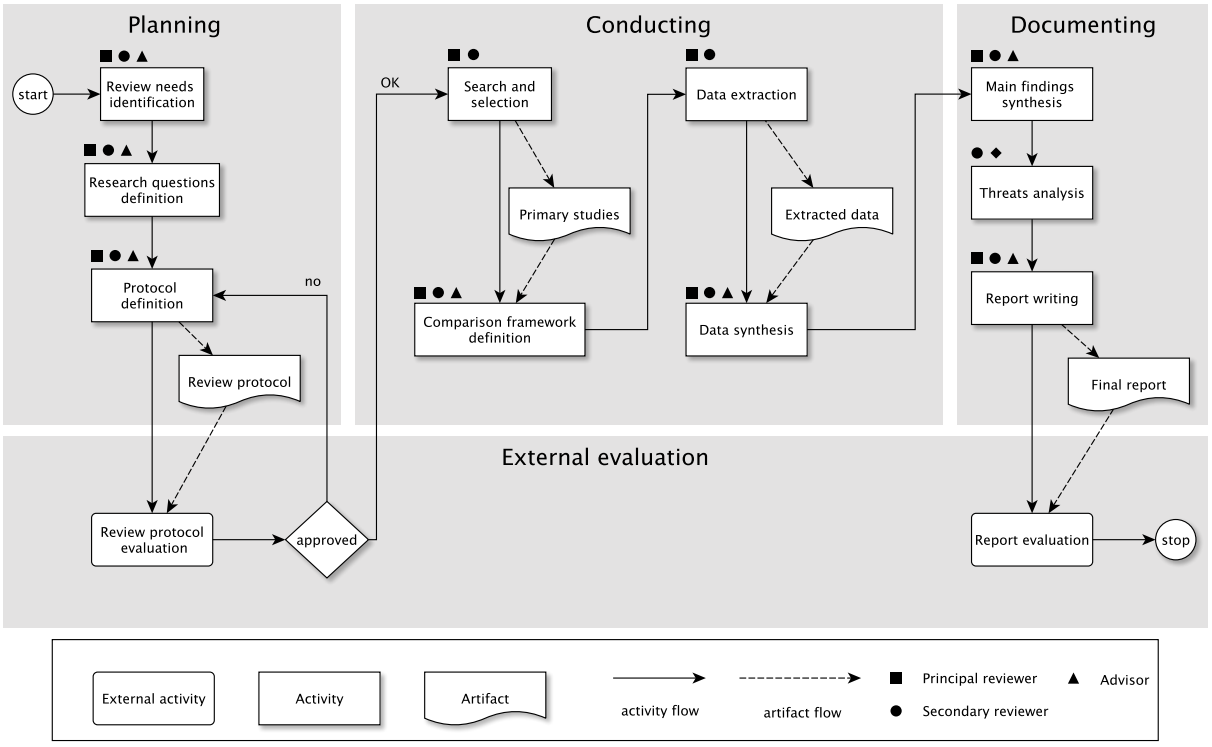

Figure 1: Overview of the whole review process

biases, some of the produced artefacts will be reviewed by external experts. More specifically, we identified two classes of external experts: SLR experts and domain experts. SLR experts are contacted for getting feedback about the proposed review protocol, possible unidentified threats to validity, possible problems in the overall construction of the review; whereas, domain experts are contacted for getting feedback about whether the proposed review protocol and final reports can be effective with respect to our topic of interest. We will contact two SLR experts and two domain experts for independent reviewing. External reviewers will have a time span of ten days to provide their feedback about the proposed artefacts.

In the following we will go through each phase of the process, highlighting main characterizing activities and produced artefacts.

2.1.1 Planning

This phase aims at (i) establishing the need for performing a review on the execution of UML models (see Section 1.2), (ii) identifying the main research questions (see Section 3), and (iii) defining the protocol to be followed by the involved researchers. The output of the planning phase is a detailed review protocol which will undergo an external evaluation by the previously identified SLR- and domain-experts and refined according to their feedback.

2.1.2 Conducting

In this phase we will perform the review by following all the steps previously defined in the review protocol. More specifically, we will carry out the following activities:

- *Search and selection*: we will perform a combination of automatic search and snowballing for identifying the comprehensive set of potentially relevant methods and techniques for the execution of UML models. Then, identified candidate entries will be filtered in order to obtain the final list of primary studies to be considered in later activities of the review. Section 4 describes in details the search and selection process.

- *Comparison framework definition*: in this activity we will define the set of parameters that will be used to compare primary studies; identified parameters will organized as a feature diagram.
- *Data extraction*: in this activity we will go into the details of each primary study, and we will fill a corresponding data extraction form. Filled forms will be collected and aggregated in order to be ready to be analyzed during the next activity. Also, in order to have a single source of information for each primary study, we will include the information coming from the quality assessment activity to the filled data extraction forms. More details about this activity are presented in Section 5.
- *Data synthesis*: this activity will focus on a comprehensive summary and analysis of the data extracted in the previous activity. The main goal of this activity is to elaborate on the extracted data in order to address each research question of the literature review (see Section 3). This activity will involve both quantitative and qualitative analysis of the extracted data. The details about this activity are presented in Section 6.

Quality evaluation is fundamental to ensure that results are based on best quality evidence [16]. It is important to note that in our study we will conduct this activity when answering a specific research question (namely, RQ3, see Section 3); indeed, in our classification framework there is a dedicated facet for the quality of evidence of existing methods and techniques for execution of UML models. The quality of each primary study will be assessed according to the criteria proposed in Section 5.3; this information will be used to (i) answer RQ3, (ii) analyze the cause of possible contradicting results, and (iii) assess the importance of individual primary studies when synthesizing results.

2.1.3 Documenting

The main activities performed in this phase are: (i) a thorough elaboration of the data extracted in the previous phase with the main aim of setting the obtained results in their context from both an academic and a pragmatic point of view, (ii) the analysis of possible threats to validity, especially to the ones identified during the definition of the review protocol (in this activity new threats to validity may emerge too), and (iii) the writing of a set of reports describing the performed literature review to different audiences (see Section 7). Firstly, produced reports will be evaluated by SLR- and domain-experts; secondly, they will be submitted to academic journals, conferences and magazines for professionals, thus they will undergo a peer reviewed evaluation by the community too.

2.2 Team

Three researchers will carry out this study, each of them with a specific role within the research team:

- *Principal researcher*: Federico Ciccozzi, senior researcher with expertise in model-driven engineering and component-based software engineering for the development of complex systems based on domain-specific modelling languages, both UML- and EMF-based. Among them, he specializes in: definition of DSMLs including UML profiles, automatic model manipulations through transformations for code generation, analysis, model optimization, system properties preservation, just to mention a few; he will be part of all the activities, i.e. planning the study, conducting it, and reporting;
- *Research methodologist*: Ivano Malavolta, post-doctoral researcher with expertise in empirical methods applied to software systems and systematic literature reviews; he will be mainly involved in (i) the planning phase of the study, and (ii) supporting the principal researcher

during the whole study, e.g., by reviewing the data extraction form, selected primary studies, extracted data, produced reports, etc.;

- *Advisor*: Bran Selic, senior researcher with many-years expertise in model-driven engineering, complex systems, UML and related profiles, simulation and execution of UML models, to mention a few. He takes final decisions on conflicts and options to 'avoid endless discussions' [18], and supports the other researchers during data and findings synthesis activities.

From a geographical point of view, the research team is entirely distributed thereby ensuring 'expertise' and reinforcing 'independence' of the individuals' reviews.

3 Research questions

When carrying out a systematic literature review, clearly defining the research questions is a pivotal task [19]. Before going into the details of the identified research questions, we formulate the goal of this research as: *to identify, classify, and evaluate existing methods and techniques for execution of UML models as deployable running artefacts*.

This goal can be refined into the following research questions. For each research question we also provide its primary objective of investigation. The research questions of this study are:

RQ1: *What are the publication trends of research studies about methods and techniques for execution of UML models?*

Objective: to classify primary studies in order to assess interest, relevant venues, and contribution types; depending on the number of primary studies, trends will be assessed over the years.

RQ2: *What are the characteristics of existing methods and techniques for execution of UML models?*

Objective: to identify and classify existing methods and techniques for executing UML models (e.g., execution strategy, considered parts of the UML family of languages, application domains, etc.).

RQ3: *What is the evidence that motivates the adoption of existing methods and techniques for execution of UML models?*

Objective: to find how much evidence is available to apply UML execution methods and techniques.

RQ4: *What are the limitations of existing methods and techniques for execution of UML models?*

Objective: to identify current gaps and limitations with respect to the state of the art in the execution of UML models as deployable running artefacts.

Answering RQ1 will give a detailed snapshot about publication trends, venues, and research groups active on the topic. Based on the classification and evaluation results obtained by answering research questions RQ2 and RQ3, we will provide a solid foundation for a thorough comparison of existing and future solutions for execution of UML models as deployable running artefacts. Finally, answering RQ4 will help the community in understanding whether there is space for improvement in this research area.

This contribution is useful for both (i) researchers to further contribute to this research area by defining new approaches or refining existing ones, and (ii) practitioners to better understand existing methods and techniques and thereby to be able to adopt the one that better suites their business goals.

The research questions listed above will drive the whole systematic review, with a special influence on (i) search and selection of primary studies, (ii) data extraction, and (iii) data analysis.

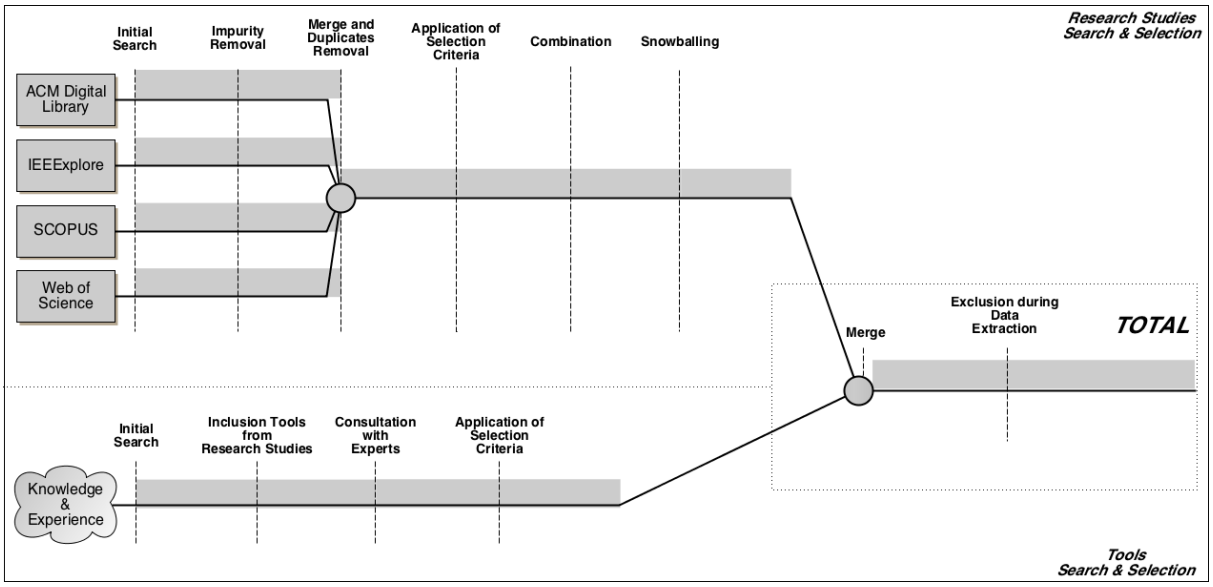

Figure 2: Search and selection process

4 Search and selection process

The main goal of our search and selection process is to retrieve both a set of research studies and tools that are relevant and representative enough for the topic being considered. Figure 2 shows our search and selection process; it has been designed as a multi-stage process in order to have full control on the number and characteristics of the studies and tools being either selected or excluded during the various stages.

According to the figure, our search and selection process is composed of two main sub-processes, each of which focusing on a specific source of information, that are:

- *Research studies search & selection*: it represents the research contributions that have been published as peer-reviewed research venues and forums, such as international journals, conferences, books, workshops, etc.;
- *Tools search & selection*: it represents the solutions proposed in terms of both open-source and commercial tools.

In both our search and selection sub-processes there is a stage in which each potentially relevant approach is evaluated against a set of selection criteria (see Sections 4.1.1 and 4.2.1). In order to handle those cases in a cost effective way we will use the adaptive reading depth [1], as the full-text reading of clearly excluded approaches will be unnecessary. Also, by following the approach proposed in [20], each potentially relevant approach will be classified by both the principal researcher and the research methodologist either as *relevant*, *uncertain*, or *irrelevant*; any approach classified as *irrelevant* will be directly excluded, whereas all the other approaches will be discussed with the help of the study advisor, if needed.

The outcomes of these sub-processes will be suitably merged in order to have one single source of information[4] for the subsequent data extraction activity. When going through a primary study in detail for extracting information, researchers can agree that the currently analysed study may be semantically out of the scope of this research, and so it shall be excluded (*Exclusion during Data Extraction*).

In the following we give a brief description of each stage of the two search and selection sub-processes.

[4]Hereafter, we will address selected research studies and tools as *primary studies*.

4.1 Research studies search and selection

Before performing the actual search and selection of relevant research studies, we manually selected a set of pilot studies. Pilot studies have been selected based on the authors knowledge of the targeted research domain (i.e., execution of UML models) and on an informal preliminary screening we performed on the available literature about the topic. Selected pilot studies fulfil the selection criteria of our study (see Section 4.1.1), they are presented in Table 1.

Title	Year
An MDE Approach for Automatic Code Generation from UML/MARTE to OpenCL [21]	2013
An Optimized Compilation of UML State Machines [22]	2012
Automatic code generation for embedded systems: From UML specifications to VHDL code [23]	2010
A co-design approach for embedded system modeling and code generation with UML and MARTE [24]	2009
eUDEVS: Executable UML with DEVS Theory of Modeling and Simulation [25]	2009
Testing and Simulating Production Control Systems Using the Fujaba Environment [26]	2000

Table 1: Pilot research studies

Pilot studies will be used to validate our search and selection strategy; basically, we will use the pilot studies to have quick feedback about the goodness of the search string we will use for our automatic search, and for guiding the definition of the selection criteria. Here the main goal is to have a search and selection criteria that at least covers all our pilot studies.
1. Initial search. In this stage we will perform automatic searches on electronic databases and indexing systems [16]. Table 2 shows the set of electronic databases and indexing systems we chose as the sources of potentially relevant studies. As suggested in [16], in order to cover as much relevant literature about UML execution methods and techniques as possible, we chose four of the largest and most complete scientific databases and indexing systems in software engineering, that are: IEEE Xplore Digital Library, ACM Digital Library, SCOPUS, and Web of Science. The selection of these electronic databases and indexing systems is guided by (i) their high accessibility, (ii) their ability to export search results to well-defined, computation-amenable formats, and (iii) the fact that they have been recognised as being an effective means to conduct systematic literature reviews in software engineering [19].

Name	Type	URL
IEEE Xplore Digital Library	Electronic database	`http://ieeexplore.ieee.org`
ACM Digital Library	Electronic database	`http://dl.acm.org`
SCOPUS	Indexing system	`http://www.scopus.com`
Web of Science	Indexing system	`http://webofknowledge.com`

Table 2: Electronic databases and indexing systems considered in this research

To create the search string, we consider (i) the research questions we defined in Section 2.1.1 and (ii) the set of pilot studies; then we extracted a list of relevant concepts, their synonyms, abbreviations, and alternative spellings, and we combined them into the final search string by using logical ANDs and ORs. The search string is shown in the listing below, it has been tested by executing pilot searches on IEEE Xplore Digital Library, and checked against all the pilot studies, which had to be part of the obtained results.

```
uml
AND (execut* OR translat* OR compilat* OR interpret* synthes* OR simulat* OR debug* OR (code
    AND generat*))
AND (mde OR mda OR mdsd OR mdd OR model* OR diagram OR specification)
```

AND *(software* ***OR*** *system* ***OR*** *tool)*

Listing 1: Search string used for automatic research studies

It is important to note that the actual search strings used in each electronic database and indexing system have been obtained by syntactically adapting them to the characteristics of each electronic database. For the sake of consistency, the search strings will be applied to title, abstract and keywords of papers in all electronic databases and indexing systems considered in this research.

Number of expected studies: around 1200

2. Impurity removal. Due to the nature of the involved electronic databases and indexing systems, search results may include also elements that are clearly not research papers, such as conference and workshop proceedings, international standards, textbooks, book series, etc. In this stage we will manually remove these results in order to have a coherent set of potentially relevant research studies.

Number of expected studies: around 1000

3. Merging and duplicates removal. In this stage all relevant results from the first stage will be combined together into a single spreadsheet. Duplicated entries will be identified and merged by matching them by `title`, `authors`, `year`, and venue of `publication`.

Number of expected studies: around 800

4. Application of selection criteria. We will consider all the selected studies and filter them according to a set of well-defined inclusion and exclusion criteria. These criteria are described in details in Section 4.1.1. In this stage, each study will be analysed in two phases: firstly it will be analysed by considering its `title`, `keywords`, and `abstract`; secondly, if the analysis will not result in a clear decision, `introduction`, and `conclusions` will be analysed too. In this stage it is crucial to select papers objectively. To this purpose, as suggested by [17], two researchers will assess a random sample of the studies, then the inter-researcher agreement will be measured using the Cohen's kappa coefficient and reported as a quality assessment of this stage in the final report. To be successful, the result of the Cohen's kappa coefficient shall be ≥ 0.80; in any case, each disagreement will be discussed and resolved with the intervention of the study advisor, if necessary. We will perform this assessment as early as possible, such that the outcome of any disagreement can be used in order to clarify the selection criteria and to make them more concrete. Under this perspective, we will perform this assessment incrementally for each database, starting from the ones with fewer results.

Number of expected studies: less than 100

5. Combination. If there will be multiple papers of the same study, we will keep a record of all of them and point them to a single study. This is necessary for ensuring completeness and traceability of our results. For example, if a primary study is published in more than one paper (for example, if a conference paper is extended to a journal version), only one instance will be counted as a primary study. Generally, the journal version would be preferred, since more complete, but both versions will be used in the data extraction phase [17].

Number of expected studies: less than 100

6. Snowballing. In our research it is fundamental that selected studies actually represent the population in terms of its ability to answer our research questions. In this context, in order to mitigate a potential bias with respect to the construct validity of the study, we will complement the previously described automatic search with a snowballing activity [27]. More in details, the main goal of this stage is to enlarge the set of potentially relevant studies by considering each study selected in the previous stages, and focusing on those papers either citing and cited by it. More technically, we will perform a closed recursive backward and forward snowballing activity [28]. From a practical point of view, we will go through each selected study by focusing on its references, and we will extract additional studies to be included in the set of relevant studies. For what concerns the forward snowballing, Google Scholar will be used to obtain those studies citing the current one [28]. In both backward and forward snowballing the initial screening of additional

studies will be based on their title only, whereas the final decision about their inclusion into the set of primary studies will be based on their abstract and title and on our selection criteria (see Section 4.1.1). Duplicates will be removed at each iteration of the snowballing activity.

Number of expected studies: less than 100

4.1.1 Selection criteria

As recommended in the guidelines for performing systematic literature reviews by Kitchenham et al. [16], the selection criteria of this study are decided during the protocol definition, so to reduce the likelihood of bias. In the following we provide inclusion and exclusion criteria for the selection of research studies. In this context, a research study will be (i) selected as a primary study if it satisfies *every* inclusion criterion, or (ii) discarded if it satisfies *any* of the exclusion criteria.

Inclusion criteria

IS1) Studies proposing methods or techniques for execution of UML models.

IS2) Studies in which UML models are (i) the main input artefacts of the proposed execution method or technique, and (ii) executed without any manual intervention.

IS3) Studies providing some kind of evaluation of the proposed method or technique (e.g., via formal analysis, controlled experiment, exploitation in industry, example usage).

IS4) Studies subject to peer review [17] (e.g., journal papers, book chapters, papers published as part of conference or workshop proceedings will be considered, whereas white papers will be discarded).

IS5) Studies written in English.

IS6) Studies available as full-text.

Exclusion criteria

ES1) Studies that do not provide any implementation of the proposed method or technique.

ES2) Studies published before 1999 (because UML was proposed in 1999).

ES3) Secondary and tertiary studies (e.g., systematic literature reviews, surveys, etc.).

ES4) Studies in the form of tutorial papers, short papers, poster papers, editorials, because they do not provide enough information.

It is important to note that, even if secondary studies will be excluded (see the E4 exclusion criteria), we will consider them in our study as follows:

- for checking the completeness of our set of primary studies (i.e., if any relevant paper will be missing from our study);
- for providing a summary of what is already known about UML models execution;
- for identifying any important issues to be considered in our study;
- for defining what is the contribution of our study to the literature.

Finally, we will collect also studies that merely empirically evaluate a method or technique in order to complement the information coming from the study actually proposing that method or technique.

4.2 Tools search and selection

Together with search and selection of research studies we will also perform a search and selection of tools providing solutions for executing UML models.

1. Initial search. In this stage we will perform a search based on (i) our direct experience with execution of UML models, (ii) specific searches for querying a generic search engine with the search string defined in Listing 4.1, and (iii) exploiting knowledge from existing networks of experts about the execution of UML models, e.g., by accessing forums, mailing lists, etc.

Number of expected tools: around 20

2. Inclusion of tools from research studies. During the search and selection for research studies we might stumble upon further tools used or referred by researchers. Those will be included in the set of tools to further process. If a included research study describes the tool's characteristics for execution of UML models, no explicit tool entry is added to the set of tools (to avoid duplication).

Number of expected tools: around 25

3. Consultation with experts. In this stage we will consult a set of experts both from industry and academia to reduce the set of selected tools to the minimal quorum so to remove, e.g., duplicates in terms of different versions of a same tool renamed during its lifetime.

Number of expected tools: around 15

4. Application of selection criteria. We will consider all the selected tools and filter them according to a set of well-defined inclusion and exclusion criteria. These criteria are presented in details in Section 4.2.1. At this point it is crucial to select tools objectively. To this end, similarly to the selection of research studies, two researchers will assess a random sample of the tools, then the inter-researcher agreement will be measured using the Cohen's kappa coefficient and reported as a quality assessment of this phase in the final report. Also in the case of tools, in order to be successful the result of the Cohen's kappa coefficient shall be ≥ 0.80; in any case, any disagreement will be discussed and resolved with the intervention of the study advisor, if necessary.

Number of expected studies: less than 10

It is important to note that in this search and selection sub-process we will focus on tools and we will collect information from their related documentation (e.g., white papers, user guides, web pages, etc.), rather than on actual scientific papers. Also, we will carry on the data extraction activity related to tools after the data extraction from research studies; doing so we will be sure that data extraction from practice may be well guided by the findings coming from the scientific papers.

4.2.1 Selection criteria

As for the research studies, also selection criteria for tools are decided during the protocol definition to reduce the likelihood of bias. A tool will be selected as a primary study if it satisfies *every* inclusion criterion, and it is discarded if it satisfies *any* of the exclusion criteria.

Inclusion criteria

IT1) Tools proposing methods or techniques for execution of UML models.

IT2) Tools in which UML models are the main input artefacts of the proposed execution method or technique.

Exclusion criteria

ET1) Tools leveraging third-party tools for executing UML models.

5 Data extraction

The main goal of this activity is to create a classification framework and a corresponding data extraction form. These two structured instruments will be used to collect data extracted from each primary study.

In our study, the classification framework will be composed of four facets, each of them addressing a research question of our research (see Section 3):

1. *Publication trends* addressing RQ1, see Section 5.1,
2. *Characteristics of methods and techniques* addressing RQ2, see Section 5.2,
3. *Evidence* addressing RQ3, see Section 5.3,
4. *Limitations* addressing RQ4, see Section 5.4,

In order to have a rigorous data extraction process and to ease the management of the extracted data, a well-structured data extraction form will be designed upfront. The form will be composed of the various categories of the classification framework. For each primary study, the principal researcher will collect in a spreadsheet a record with the extracted information for subsequent analysis: the spreadsheet columns will be the parameters, while each spreadsheet row will represent the data of each primary study.

As suggested in [17], the principal and the research methodologist will pilot the data extraction form independently. In order to validate our data extraction strategy, we will perform a sensitivity analysis to check whether the results are consistent independently from the researcher performing the analysis. More specifically, the principal researcher and the research methodologist will get a random sample of 10 primary studies and both of them will analyze them independently by filling the data extraction form for each primary study. So, in order to make the whole study more reliable, two researchers will read each primary study of the random sample and extract information from it. Then, the Cohen's kappa coefficient will be applied to the obtained results to assess the level of agreement among the researchers. The value of the obtained Cohen's kappa coefficient will be documented in the final report. Finally, each disagreement will be discussed and resolved with the intervention of research advisor, if necessary.

5.1 Publications trends (RQ1)

In the following we list all the parameters we will consider to collect data about publication and research trends on execution of UML models, together with some examples of analysis we will perform:

1. *Publication year*: for extracting the publication tendency per year;
2. *Publication venue*: it will be analysed in terms of their distribution, and also with respect to their distribution over time. These are the types of publication venues we will consider:
   (a) journal paper,
   (b) book chapter,
   (c) conference paper,
   (d) workshop paper;
3. *Authors*: by using the authors as index, we will identify (i) how many different authors have been working on the topic, and (ii) how many primary studies each author has been working on during the years. We will perform the same analysis for identifying prominent research groups.

4. *Application fields*: we will identify the distribution of primary studies with respect to the application field where the proposed methods and techniques have been applied (e.g., automotive, web applications, real-time embedded systems). In this case we will reuse the taxonomy of software application types proposed by Forward and Lethbridge [29].

5. *Research strategies*: since this facet is general and independent from a specific research area, we reuse the classification of research approaches proposed by Wieringa et al. in [30]. We chose this classification because (i) it has been widely used in various systematic mapping studies (e.g., in [31–33]), and (ii) its categories are quite cost-effective to be identified by reading a paper without going into its very details [1]. In Table 3 we report the classification proposed by Wieringa et al. According to this classification, papers can span more than one category, although some combinations are unlikely [30].

Category	**Description**
Validation Research	Techniques investigated are novel and have not yet been implemented in practice. Techniques used are for example experiments, i.e., work done in lab.
Evaluation Research	Techniques are implemented in practice and an evaluation of the technique is conducted. That means, it is shown how the technique is implemented in practice (solution implementation) and what the consequences of the implementation in terms of benefits and drawbacks (implementation evaluation) are. This also includes identification of problems in industry.
Solution Proposal	A solution for a problem is proposed, the solution can be either novel or a significant extension of an existing one. The potential benefits and the applicability of the solution is shown through a small example or a good line of argumentation.
Philosophical Papers	These papers sketch a new way of looking at existing things by structuring the field in form of a taxonomy or conceptual framework.
Opinion Papers	These papers express the personal opinion of somebody whether a certain technique is good or bad, or how things should been done. They do not rely on related work nor research methodologies.
Experience Papers	They explain on what and how something has been done in practice. It has to be the personal experience of the author.

Table 3: Research classification (extracted from [1])

5.2 Characteristics of methods and techniques (RQ2)

Since research question RQ2 is at the core of our research, the creation of its corresponding facet in the data extraction form demands a detailed analysis of the contents of each primary study. In light of this, we will follow a systematic process called *keywording* for defining the categories of this facet of our classification framework. Goal of the keywording process is to effectively develop a classification framework so that it fits existing studies and tools and takes their characteristics into account [1]. Moreover, from the first analysis of the pilot studies we noticed that there is a large variability among the various methods and techniques for execution of UML models, so we will refine the classification framework during the data extraction activity, if needed.

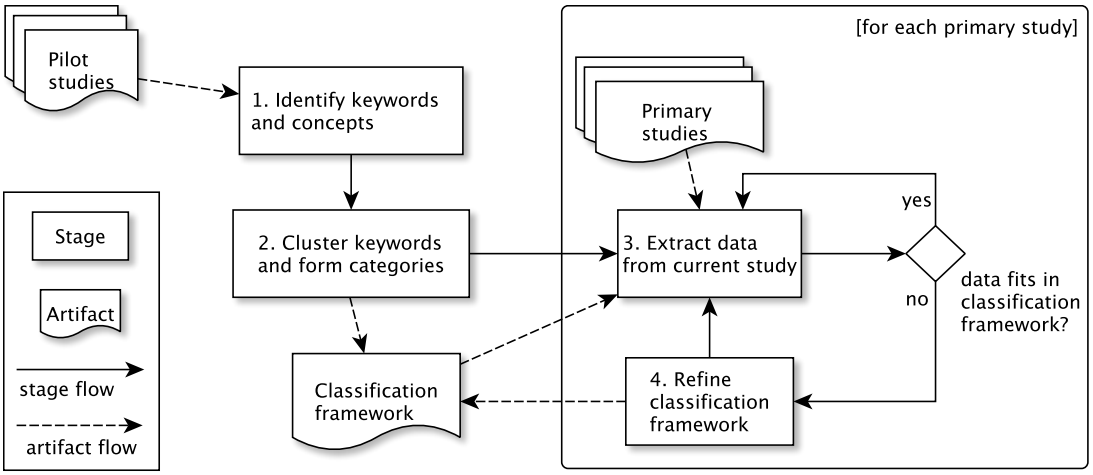

Figure 3: Keywording process and data extraction for RQ3

Figure 3 shows the process we will follow for defining the facet of our classification framework for answering RQ3. In the following we detail each step of the process depicted in the figure:

1. *Identify keywords and concepts*. Principal researcher and research methodologist will collect keywords and concepts by reading the full-text of each pilot study. When all pilot studies have been analyzed, all keywords and concepts are combined together to clearly identify the context, nature, and contribution of the research about execution of UML models. Bearing in mind that the researchers and the authors of the primary studies may use different terms for same concepts and vice versa, we will collate different keywords and terms to ensure consistency and compatibility. The output of this stage is the set of keywords and concepts extracted from the pilot studies.

2. *Cluster keywords and form categories*. Principal researcher and research methodologist will perform a clustering operation on collected keywords and concepts in order to cluster them according to the emerging categories. The output of this stage is the initial classification framework containing all the identified categories, each of them representing a specific aspect of a method or technique for execution of UML models. In this step we expect that emerging categories may be: execution strategy (i.e., interpretation, compilation, or code generation), considered subset of UML, types of auxiliary models other than the UML ones, application domain, method goal (e.g., improving performance of the system, platform independency, speed up of development process, etc.). Next steps will be performed for each primary study.

3. *Extract data from current study*. In this step the principal researcher will extract information about the current primary study to be analysed and will (i) collect information according to the categories of the classification framework and (ii) collect any kind of additional information that is considered relevant and that does not fit within any category of the classification framework. If the collected information about the current primary study fits completely within the classification framework, then the principal researcher can analyze the next primary study, otherwise the classification framework must be refined.

4. *Refine classification framework*. In this step the principal researcher and research methodologist will discuss about the collected additional information that does not fit into any category of the current classification framework. This discussion may reveal that:

   - the principal researcher did not interpret correctly the collected information. In this case the collected information will be correctly classified according to the classification framework;

- the categories of the current classification framework are not representative enough for the considered primary study. In this case the classification framework will be refined so that it will better fit the collected information; additionally, previously analyzed primary studies will be re-analyzed according to the refined classification framework.

The above described process will end when there will be no primary studies to analyze left.

5.3 Evidence (RQ3)

This facet of our classification framework can be divided into three different sub-facets, that are about: what is the actual evidence, its rigor, and its industrial relevance.

For what concerns the first sub-facet, we will collect relevant fragments of either a primary research study or about the documentation of an tool; depending on the type of collected information, it will be suitably analyzed via qualitative analysis methods (e.g., coding, thematic analysis, etc.).

For what concerns rigor and industrial relevance, we will apply the method for evaluating rigor and industrial relevance of technology evaluations proposed by Ivarson and Gorscheck [34].

More specifically, *rigor* focusses on the extent to which aspects related to the actual rigor of a study are presented; the main rationale behind this is that a study with an high rigor score makes the evaluation by reviewers or readers easy and straightforward. Based on [34], the rigor of each study will be assessed according to the criteria in Table 4, where all criteria can be scored with the following score levels: strong (1 point), medium (0.5 point), weak (0 points). Thus, a primary study can have a rigor score ranging from 0 to 3, depending on its rigor level.

ID	Criteria Name	Rigor criteria
R1	Context	Is the context described to the degree where a reader can understand and compare it to another context?
R2	Study design	Is the study design described to the degree where a reader can understand its main parts, e.g., variables, treatments, etc.?
R3	Validity	Is the validity and threats of the study discussed and measured in details?

Table 4: Rigor assessment criteria

Industrial relevance refers to the realism of the evaluation of a method or technique, and determines the potential relevance of its results for industry. Table 5 shows the criteria we will use for assessing the industrial relevance of each primary study. According to [34], each industrial relevance criteria will be assessed by assigning a numerical value to it, similarly to the way in which the rigor score will be assessed.

ID	Criteria Name	Industrial relevance criteria
IR1	Subjects	Are subjects used in the evaluation representative?
IR2	Context	Is the evaluation performed in a representative setting?
IR3	Scale	Is the scale of the applications used in the evaluation of realistic size?
IR4	Research method	Does the applied research method facilitate the investigation of real situations?

Table 5: Industrial relevance assessment criteria

5.4 Limitations (RQ4)

This facet of our classification framework will focus on the identification of limitations regarding the various methods and techniques for execution of UML models. Accordingly, in our classification

framework we will have a dedicated category called *limitations* in which the principal researcher will collect all the limitations and unsolved challenges discussed in the various primary studies. Also, we will combine this information with the answers to previous research questions (i.e., RQ1, RQ2, and RQ3) in order to identify further limitations with respect to the different concerns raised by the other research questions.

6 Data synthesis

The data synthesis activity involves collating and summarising the data extracted from the primary studies [7, § 6.5] with the main goal of understanding, analysing, and classifying current research on the execution of UML models.

In this phase we will have a fully populated spreadsheet with all the information coming from the data extraction form of each primary study. According to this, our data synthesis will be divided into two main phases: vertical analysis and horizontal analysis. When performing *vertical analysis*, we will analyze the extracted data to find trends and collect information about each parameter of each category of our classification framework. When performing *horizontal analysis*, we will analyse the extracted data to explore possible relations across different parameters of our research. In both phases we will perform a combination of content analysis (mainly for categorizing and coding approaches under broad thematic categories) and narrative synthesis (mainly for explaining in details and interpreting the findings coming from the content analysis).

Vertical analysis. Depending on the parameters of the classification framework (see Section 5), in this research we will apply both quantitative and qualitative synthesis methods, separately. When considering quantitative data, depending on the specific data to be analysed, we will perform a specific kind of descriptive statistics for better understanding the data about a specific parameter. When considering qualitative data, we will apply the *line of argument* synthesis [17], that is: firstly we will analyse each primary study individually in order to document it and tabulate its main features with respect to each specific parameter of the classification framework defined in Section 5, then we will analyse the set of studies as a whole, in order to reason on potential patterns and trends. When both quantitative and qualitative analyses are completed, we will integrate their results in order to explain quantitative results by using qualitative results [7, § 6.5].

Horizontal analysis. We will cross-tabulate and group the data, and make comparisons between two or more nominal variables. The main goal of the horizontal analysis is to (i) investigate on the existence of possible interesting relations between data pertaining to different facets of our research. In this context, we will use cross-tabulation as strategy for evaluating the actual existence of those relations.

Independently of the performed data synthesis analysis, we will perform a **sensitivity analysis** to check whether the obtained results are consistent across different subsets of studies. Examples of subsets of primary studies are:

- high quality primary studies only;
- low quality primary studies only;
- primary studies performing code generation only;
- primary studies with in-the-field validation only (e.g., real-world benchmarking, industrial case study, etc.);
- primary studies published in the last three years only.

7 Dissemination strategy

We plan to report our systematic study to different audiences, to reach both researchers and practioners. In the following we list the actions we will undertake in our dissemination strategy:

1) we will report our main research-oriented findings and a detailed description of this study into an *research publication* in an international research journal. Possible targets are: Information and Software Technology journal[5] (IST) or Transactions on Software Engineering[6] (TSE);

2) an accompanying *technical report* will present all the details and raw data of the study; the chief aim of the technical report is to make our study reproducible by interested researchers and to foster its replication;

3) we will target a *practitioners-oriented magazine*, with the goal of impacting and enhancing the current state of the practice about execution of UML models.

8 Review schedule

Activity	**Deadline**	**Assigned to**
Defined protocol	18 Mar 2015	Ivano and Federico
Evaluated and refined protocol	15 May 2015	Ivano, Federico, Bran, and external evaluators
Search complete	30 May 2015	Ivano and Federico
Selection complete	10 June 2015	Ivano and Federico
Classification framework complete	20 July 2015	Ivano, Federico, and Bran
Data extraction complete	3 Dec 2015	Ivano and Federico
Data synthesis complete	30 Jan 2016	Ivano and Federico
Findings extraction and report writing	30 Feb 2016	Ivano, Federico, and Bran

[5] http://www.journals.elsevier.com/information-and-software-technology/
[6] http://www.computer.org/web/tse

References

[1] 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, British Computer Society, Swinton, UK, UK, 2008, pp. 68–77.
URL http://dl.acm.org/citation.cfm?id=2227115.2227123

[2] D. C. Schmidt, Guest editor's introduction: Model-driven engineering, Computer 39 (2) (2006) 25–31.

[3] K. Czarnecki, S. Helsen, Feature-based survey of model transformation approaches, IBM Systems Journal (2006) 621–645.

[4] I. Malavolta, P. Lago, H. Muccini, P. Pelliccione, A. Tang, What industry needs from architectural languages: A survey, IEEE Trans. Software Eng. 39 (6) (2013) 869–891.
URL http://doi.ieeecomputersociety.org/10.1109/TSE.2012.74

[5] A. Abouzahra, J. Bézivin, M. D. Del Fabro, F. Jouault, A practical approach to bridging domain specific languages with uml profiles, in: Proceedings of the Best Practices for Model Driven Software Development at OOPSLA, Vol. 5, Citeseer, 2005.

[6] F. J. Lucas, F. Molina, A. Toval, A systematic review of uml model consistency management, Information and Software Technology 51 (12) (2009) 1631–1645.

[7] Tech. rep.

[8] D. Budgen, A. J. Burn, O. P. Brereton, B. A. Kitchenham, R. Pretorius, Empirical evidence about the uml: a systematic literature review, Software: Practice and Experience 41 (4) (2011) 363–392.

[9] M. G. H. J. Nelson, M. Piattini, A systematic literature review on the quality of uml models, Innovations in Database Design, Web Applications, and Information Systems Management (2012) 310.

[10] B. Dobing, J. Parsons, How uml is used, Communications of the ACM 49 (5) (2006) 109–113.

[11] T. Gorschek, E. Tempero, L. Angelis, On the use of software design models in software development practice: An empirical investigation, Journal of Systems and Software 95 (2014) 176–193.

[12] T. Gorschek, E. Tempero, L. Angelis, A large-scale empirical study of practitioners' use of object-oriented concepts, in: Software Engineering, 2010 ACM/IEEE 32nd International Conference on, Vol. 1, IEEE, 2010, pp. 115–124.

[13] M. Usman, A. Nadeem, T.-h. Kim, E.-s. Cho, A survey of consistency checking techniques for uml models, in: Advanced Software Engineering and Its Applications, 2008. ASEA 2008, IEEE, 2008, pp. 57–62.

[14] E. Domínguez, J. Lloret, B. Pérez, Á. Rodríguez, Á. L. Rubio, M. A. Zapata, A survey of uml models to xml schemas transformations, in: Web Information Systems Engineering–WISE 2007, Springer, 2007, pp. 184–195.

[15] K. Czarnecki, S. Helsen, Feature-based survey of model transformation approaches, IBM Systems Journal 45 (3) (2006) 621–645.

[16] B. Kitchenham, P. Brereton, A systematic review of systematic review process research in software engineering, Information and software technology 55 (12) (2013) 2049–2075.

[17] C. Wohlin, P. Runeson, M. Höst, M. Ohlsson, B. Regnell, A. Wesslén, Experimentation in Software Engineering, Computer Science, Springer, 2012.

[18] H. Zhang, M. A. Babar, Systematic reviews in software engineering: An empirical investigation, Information and Software Technology 55 (7) (2013) 1341–1354.

[19] P. Brereton, B. A. Kitchenham, D. Budgen, M. Turner, M. Khalil, Lessons from applying the systematic literature review process within the software engineering domain, Journal of Systems and Software 80 (4) (2007) 571–583.

[20] N. B. Ali, K. Petersen, Evaluating strategies for study selection in systematic literature studies, in: International Symposium on Empirical Software Engineering and Measurement, ACM, 2014.

[21] A. Rodrigues, F. Guyomarc'h, J.-L. Dekeyser, An mde approach for automatic code generation from uml/marte to opencl, Computing in Science Engineering 15 (1) (2013) 46–55.

[22] A. Charfi, C. Mraidha, P. Boulet, An optimized compilation of uml state machines, in: Object/Component/Service-Oriented Real-Time Distributed Computing (ISORC), 2012 IEEE 15th International Symposium on, IEEE, 2012, pp. 172–179.

[23] T. Moreira, M. Wehrmeister, C. Pereira, J. Petin, E. Levrat, Automatic code generation for embedded systems: From uml specifications to vhdl code, in: Industrial Informatics (INDIN), 2010 8th IEEE International Conference on, 2010, pp. 1085–1090.

[24] J. Vidal, F. de Lamotte, G. Gogniat, P. Soulard, J.-P. Diguet, A co-design approach for embedded system modeling and code generation with uml and marte, in: Design, Automation Test in Europe Conference Exhibition, 2009. DATE '09., 2009, pp. 226–231.

[25] J. L. Risco-Martín, J. M. De La Cruz, S. Mittal, B. P. Zeigler, eudevs: Executable uml with devs theory of modeling and simulation, Simulation 85 (11-12) (2009) 750–777. doi:10.1177/0037549709104727.
URL http://dx.doi.org/10.1177/0037549709104727

[26] J. Niere, A. Zndorf, Testing and simulating production control systems using the fujaba environment, in: M. Nagl, A. Schrr, M. Mnch (Eds.), Applications of Graph Transformations with Industrial Relevance, Vol. 1779 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2000, pp. 449–456. doi:10.1007/3-540-45104-8_37.
URL http://dx.doi.org/10.1007/3-540-45104-8_37

[27] T. Greenhalgh, R. Peacock, Effectiveness and efficiency of search methods in systematic reviews of complex evidence: audit of primary sources, BMJ 331 (7524) (2005) 1064–1065.

[28] C. Wohlin, Guidelines for snowballing in systematic literature studies and a replication in software engineering, in: Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering, EASE '14, ACM, New York, NY, USA, 2014, pp. 38:1–38:10.

[29] A. Forward, T. C. Lethbridge, A taxonomy of software types to facilitate search and evidence-based software engineering, in: Proceedings of the 2008 conference of the center for advanced studies on collaborative research: meeting of minds, ACM, 2008, p. 14.

[30] R. Wieringa, N. Maiden, N. Mead, C. Rolland, Requirements engineering paper classification and evaluation criteria: a proposal and a discussion, Requirements Engineering 11 (1) (2006) 102–107. doi:10.1007/s00766-005-0021-6.
URL http://dx.doi.org/10.1007/s00766-005-0021-6

[31] E. Engström, P. Runeson, Software product line testing - a systematic mapping study, Inf. Softw. Technol. 53 (1) (2011) 2–13. doi:10.1016/j.infsof.2010.05.011.
URL http://dx.doi.org/10.1016/j.infsof.2010.05.011

[32] A. Mehmood, D. N. Jawawi, Aspect-oriented model-driven code generation: A systematic mapping study, Information and Software Technology 55 (2) (2013) 395 – 411, special Section: Component-Based Software Engineering (CBSE), 2011. doi:http://dx.doi.org/10.1016/j.infsof.2012.09.003.
URL http://www.sciencedirect.com/science/article/pii/S0950584912001863

[33] K. Petersen, Measuring and predicting software productivity: A systematic map and review, Information and Software Technology 53 (4) (2011) 317 – 343, special section: Software Engineering track of the 24th Annual Symposium on Applied Computing Software Engineering track of the 24th Annual Symposium on Applied Computing. doi:http://dx.doi.org/10.1016/j.infsof.2010.12.001.
URL http://www.sciencedirect.com/science/article/pii/S0950584910002156

[34] M. Ivarsson, T. Gorschek, A method for evaluating rigor and industrial relevance of technology evaluations, Empirical Software Engineering 16 (3) (2011) 365–395.