Protocol for a systematic mapping study on Model-driven Engineering for Mobile Robotic Systems

Giuseppina Lucia Casalaro, Giulio Cattivera, Federico Ciccozzi,
Ivano Malavolta, Andreas Wortmann, Patrizio Pelliccione

1. Introduction

This document describes the review protocol of a systematic mapping study aimed at identifying, evaluating and classifying existing research in MDE approaches for mobile multi-robot systems.

2. Background

Since the 2000s the presence of software has significantly increased so that nowadays companies that produce software are among the largest and most prolific in business. One of the main reason is that software covers a primary role within most of the technologies and services of nowadays. It is therefore true that for those companies that produce software, an high cost of development process or the presence of inadequacies in the architecture and design can cause serious economic issues [1]. Among the major causes for bad software and high costs of production, there is the complexity of the software itself. The intricacy of paradigms and criteria upon which the structure of the software is based, can bring serious issues both during the development and the maintenance phase of a system. In other words, a complexity not properly handled, has always meant financial losses for companies that produce complex software systems.

During time, engineers have always dealt with software complexity by inventing mechanisms of abstraction. Giving the programmer the ability to abstract from the underlying environment basically means to simplify the complexity of software. In fact the most is the abstraction from the platform details, the highest is the focus of the developer on his design intent rather than on surrounding platform/technology details.

In history it is possible to find numerous examples: the first programming languages such as Assembly shielded developers from the details of the machine code, the Operating Systems provided abstraction from the hardware so that developers did not have to programme directly on the hardware.

Strides have been increasingly made until in 1980's Computer-aided Software Engineer (CASE) was introduced. Its aim was essentially to express programs with graphics elements such as state machine diagrams, structure diagrams, flow diagrams and so on in order to reason on the goals without distraction from implementative platform details. One of the purposes of CASE was also to generate code directly from the graphical representations. But it was not that much used in practice because the mapping between objects and code was poor and the supported operating systems were single node, missing of several quality of service properties (e.g., security, fault tolerance etc).

More recently, high-level programming languages such as Java, C++ or C# have been introduced and they have strongly raised the level of abstraction from platforms. However nowadays the number of new platforms and their complexity are increasing, much more faster than the capacity of these languages to mask the complexity. For example a way to simplify the platform complexity with Java and C# is represented by middleware platforms like J2EE and .NET, but on the other hand their structure is so intricately of dependencies that subtle effects always appear.

However, when it comes to large-scale complex systems composed of hundreds or thousands of software components, the structure is never so immediate to the point of being able to visualize and immediately implementing it with whatever high-level programming language.

The software of a system has its inner construction paradigms - criterions, that establish the structure of the system. It is not possible to recognize the suitable construction paradigms of large systems at the level of programming languages, because the capacity of abstraction is too much low.

2.1 Models

An abstraction mechanism suitable for simplifying the complexity of large software systems is represented by models. Using models it is possible to represent systems in an efficient and concise way so that developers have a reference to the criterions that must be used to match the desired system. In other words, developers can constantly check during the development phase if their code is adhering to the designed structure of the system. For these reasons it is true that models for describing systems and their sub-parts, have always been used and represent an enduring tradition in Software Engineering.

However, the role of models in traditional Software Engineering methodologies presents at least two noteworthy disadvantages. Firstly, they relegate the role of models to a mere documentation, thus the relation between models and code is not formal but it is left to the intention of the developer. In fact, developers generally start from scratch and have to comply the adherence with models by constantly doing manual consistency checks between code and models. This is because it must be avoided the risk of going out of the criterions of the designed structure and generating local or distributed problems within the whole system [1].

The second disadvantage concerns the maintainability of models and code. Since system architectures and design are not static, they may be changed during several stages of the life-cycle. Thus, the documentation of models must be scrupulously updated and for large systems this can represent a tricky task [1]. Moreover, after a change in the model, the code must be kept consistent with the new modifications and in traditional Software Engineering this task is done by hand.

This way of proceeding may itself require long production times and if the risks discussed above are not properly considered, there could be even more delay in the release of the final product and a consequently increase of production costs.

2.2 Mobile Multi-Robot Systems and Model-Driven Engineering

Since the ’70s, the presence of robots in industry had become more and more massive. The most typical robots are robotic arms called manipulators (or assembly robots), they are fixed on a surface and they are able to accomplish tasks with high speed and accuracy. For instance, in the electronics industry, it is the presence of these robots that makes possible products such as computers and smartphones.

However, the biggest limitation for these type of robots is the lack of freedom in terms of movement. In fact, they are attached on a fixed plan and thus they have a limited range of action. On the other hand, there exists another category of robot systems defined as ”Mobile Robot Systems” (MRSs). As the name suggests, they are able to move freely in the space, that is, going around the space in a non-supervised manner to accomplish their tasks [2]. Due to this ability, the field of application for these robots is wide, a mobile robot can be used not only in the industrial production but also in the services. For example, the robot ”AVG” (Autonomous Guided Vehicle) independently carries parts between different production stations. Moreover, the ”Helpmate service robot” carries medicines between hospitals. Then there are also robots that explore planets themselves, or simply there are robots that are used to clean large buildings or uncomfortable places like subways.

However, when a multiple number of mobile robots are put together to accomplish a group task, a new category of robots is identified: mobile multi-robots. Thus, a system of such robots is called “Mobile Multi-robot Systems” (MMRSs). In MMRSs robots cooperate in team, each robot has its specific task in order to achieve the overall mission assigned to the entire team. Using multiple robots together opens for several new opportunities. In fact, there are missions that requires group characteristics and cannot be accomplished by a single robot, moreover a team can realize missions much more quickly than a single robot. [9]

In the last years, MMRSs have become more and more anthropomorphic and their use as well as the related research, is continuously increasing. Nowadays MMRSs are so mainstream that it is reasonably expected that future will be pervaded by robots [9]. Underwater, terrain, and flying robots will simplify the everyday life of people and will add plenty of new opportunities of applications.

Software development for MMRSs is still very complex and nowadays there is not yet a standard neither in terms of methodology nor in terms of platforms. Although there have been made several attempts and proposals, an agreement has not yet been reached. The most relevant limitation that was found in these attempts is that they were strongly dependent on the underlying platform, programming languages and interfaces of the various components of the robot [3]. In fact, in a MMRS there are sensors, actuators, artificial intelligence modules, network devices and each of them is implemented as a single component that needs an appropriate framework, programming language and interfaces [3]. The traditional software engineering methodologies that have been used so far were too much influenced by this componentization. Being so strongly influenced by the componentization is not desirable in MMRSs, because it is in their inner nature to be highly multidisciplinary and this cannot be changed, thus we want software engineering methodologies that can efficiently manage this aspect and overcome the related issues. We want a methodology that permits abstracting from the nature and structure of the robot components and that can efficiently overcome issues of heterogeneity. Heterogeneity issues are always present in high componentized and multidisciplinary systems like the MMRSs and these strongly affect the production of code. This happens because MMRSs are always developed with methodologies that provide for the componentization of modules, where each of them is developed using its appropriate technology, framework, patterns and programming language. This led to problems of interoperability among modules but also among different MMRSs [3].

The need for software engineering methodologies that can efficiently support developing and maintaining of MMRSs is high. Currently Model-Driven Engineering (MDSE or MDE) seems a very effective methodology for the development of MMRSs: it would simplify the complexity of robots by automatically generating code from models defined with domain concepts and not based on platform and technology.

Model-driven Software Engineering

MDE can be defined as a methodology for applying the advantages of modeling to Software Engineering activities [10]. *"Everything is a model"* is the paramount principle of MDE according to which MDE aims at manage, guide and found the process of developing a software system on the concept of *model*. In other words, the aim of MDE is to create formal models to represent systems and from which it is possible to generate the majority of code of the final implementation [1]. Thus, all MDE methods and techniques consider the logical role of models at the same level of the code.

In addition to models, the second fundamental concept in MDE is the one of *transformation*. A transformation is an operation that can be performed on one or more models in a certain domain. In other words, all the desired manipulations on models must be formally defined through transformations. Thus, both the definitions of models and transformations should be done throughout some notation, i.e. throughout *modeling languages*. The latter may be either general-purpose languages or domain-specific languages. However, regardless of the type of language, the ethic of MDE is to formalize and to define models according only to the domain of the system, and not according to the programming language that will be used for the final implementation [10].

Moreover, the definition of a modeling language itself is considered as a model in MDE. In fact, this procedure is called metamodeling, with which it is actually modeled a modeling language. Theoretically, this process can be recursive: in fact one can model a metamodel, which means to define a meta-metamodel. All MDE approaches share this ethic in all the fields of application [1].

One of the biggest mistake which is commonly done when talking about MDE, is the one of associating the action of modeling to the simple activity of drawing graphical objects, that can possibly be in compliance with some syntactic rules. Instead, in MDE the action of modeling has a much wider meaning, in fact with models it's possible to carry out actions like syntactical validation, model checking, model simulation, model transformations, model execution either through code generation or model interpretation, and model debugging. All these possibilities are meant to simplify the development and maintainability of the system, since both the creation and manipulations of models can be automatically reported on the real system. Thus, it goes without saying that this is something that can not be made from mere graphical drawn objects [10].

Figure 1 summarizes the fundamental ingredients of MDE and their connections:

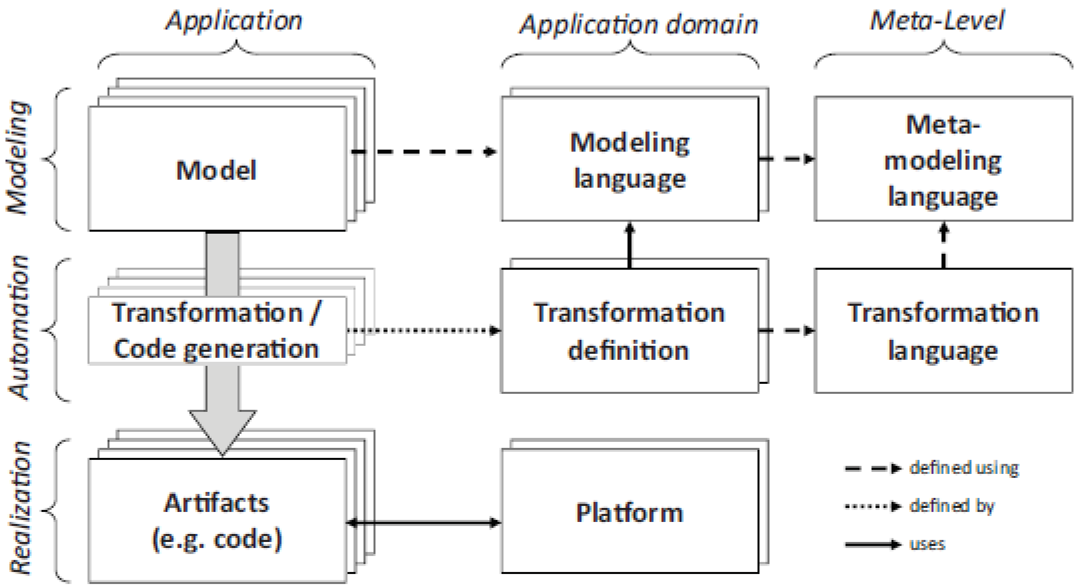

Figure 1: Overview of the MDSE methodology [10]

The main flow of events that represents the ethics of MDE is the one that goes from the application model down to the implementation. As shown in Figure 1, this flow contains transformations and code generation. These latter are the step that reflects one of the main advantages of MDE: *reuse the models to generate code on different platforms*. The realization level is the software that actually runs on a certain platform, if the platform changes the new code will be generated starting from the same models that can possibly be subject to prior transformations. This is possible thanks to the definition of models and transformations performed with modeling languages, that in turn are defined by metamodeling languages [10]. It goes without saying that the concept of modeling language is one of the main ingredients of the MDE and it's worth spending some more explanation.

A modeling language is a tool that allows to specify models for a certain domain of systems. There are two types of modeling language: Domain-Specific Modeling Languages (DSMLs) and General-Purpose Modeling Languages (GPMLs, GMLs, GPLs).

Domain-Specific Modeling Languages: they are languages defined for modelling only things of a specific domain in order to enhance the task of people involved in the creation of things of that context.

General-Purpose Modeling Languages: they represent modeling tools that can be used in any domain. The typical examples are the UML language or Petri-nets.

In a practical sense, metamodels basically constitute the definition of a modeling languages, since they provide a way of describing the whole class of models that can be represented by that language. As mentioned, this process can be recursive, but in practice it is seen that the metamodels typically can be defined by themselves and therefore does not make much sense to go beyond this level of abstraction.

As regards transformations of models, they are also defined through formal languages that are called transformation languages and that provide everything needed to create transformation rules. Transformations are performed between a source and a target model, but it is actually defined upon the respective metamodels [10].

2.3 The need for a systematic mapping study on the topic

Both MDE and MMRSs are new, as a consequence proprietary MDE frameworks and infrastructures for developing MMRSs that have been proposed, are not yet standard approaches since they focus on particular type of components and functions of domain-specific robots [3]. Thus, a methodical standard use of MDE for MMRSs is still missing.

Moreover we have found that at the time of writing, in literature there is a lack of secondary researches that can illustrates the state-of-the-art for MDE applied to MMRSs, so researchers do not yet have an exhaustive view of such a new thing. Thus, we want to perform a systematic mapping study on that topic, in order to help both researchers to get a clearer view on the state-of-the-art for MDE applied to MMRSs, and practitioners to understand to what extent this methodology has been used for real-world software projects. Hence, through our work researchers can get a large overview on the topic, identify limitations, successes and weaknesses, in order to efficiently continue the research. While practitioners will understand to what extent this methodology is suitable for real-world software projects by looking at the information from the previous real applications that we will show.

Since MDE on robotics is a quite recent field, identifying the various publications venues and trends is an indispensable step towards a fully understanding of this research area. In fact in accordance with [4] a Systematic map should be used as a first step toward a systematic review, first the topic area is structured and thereafter a specific focus area is investigated with a systematic review.

3. Research implementation

3.1 Process

We define the process of systematic mapping study following the guidelines defined in [5]. The Figure 2 shows the overview of the whole process of our research.
Three well-established phases of a systematic mapping study in software engineering are defined as follows in accordance with [4]:

- *Planning* the study by structuring a formal protocol
- *Conducting* the study by collecting, analysing, and classifying relevant research on the topic
- *Reporting* and discussing the findings

The *planning phase* begins with the identification of the motivation and the need of a systematic mapping study. Subsequently, the research questions are defined following the PICOC guidelines from [5]. During the planning phase, the protocol development can be improved through an iterative loop or refinements that might occur after the evaluation step. When no more reasonable refinements are necessary, then a document containing the current version of the protocol is produced.

The *conducting phase* represents the mapping study, and it includes the following steps:

- *Search for studies:* in this step all studies are collected;
- *Selection of primary studies:* all initial studies are reduced by applying the inclusion and exclusion criteria. At the end of this step the set of primary studies are defined;
- *Classification scheme definition:* allows the definition of the classification scheme. At the end, the data extraction form is produced which is a spreadsheet that will be filled in the next step;
- *Data extraction:* In this step the data extraction form is filled. It is based on the classification scheme, and it is a spreadsheet in which the columns represent the attributes defined by facets, and each row represent a study;
- *Data analysis:* in this last step the primary studies are analysed in order to answer the research questions. If the analysis of the extracted data confirms the appropriateness of the classification scheme, then the creation of the systematic map is considered to be completed, otherwise an iterative process of refinements is carried out in order to improve the classification scheme.

The last *reporting phase* involves the systematic discussion on the map and the writing of the report of the entire study.

To sum up, we will first define the core of the entire mapping study that is the protocol. Basically the protocol of a mapping study formalises the process of the study. With the protocol, we define the rules by which the study is carried out, the way in which research literature is selected, the way in which information from literature is extracted, the way in which results are presented and also we define procedures to validate and verify the study strategy and the obtained results.

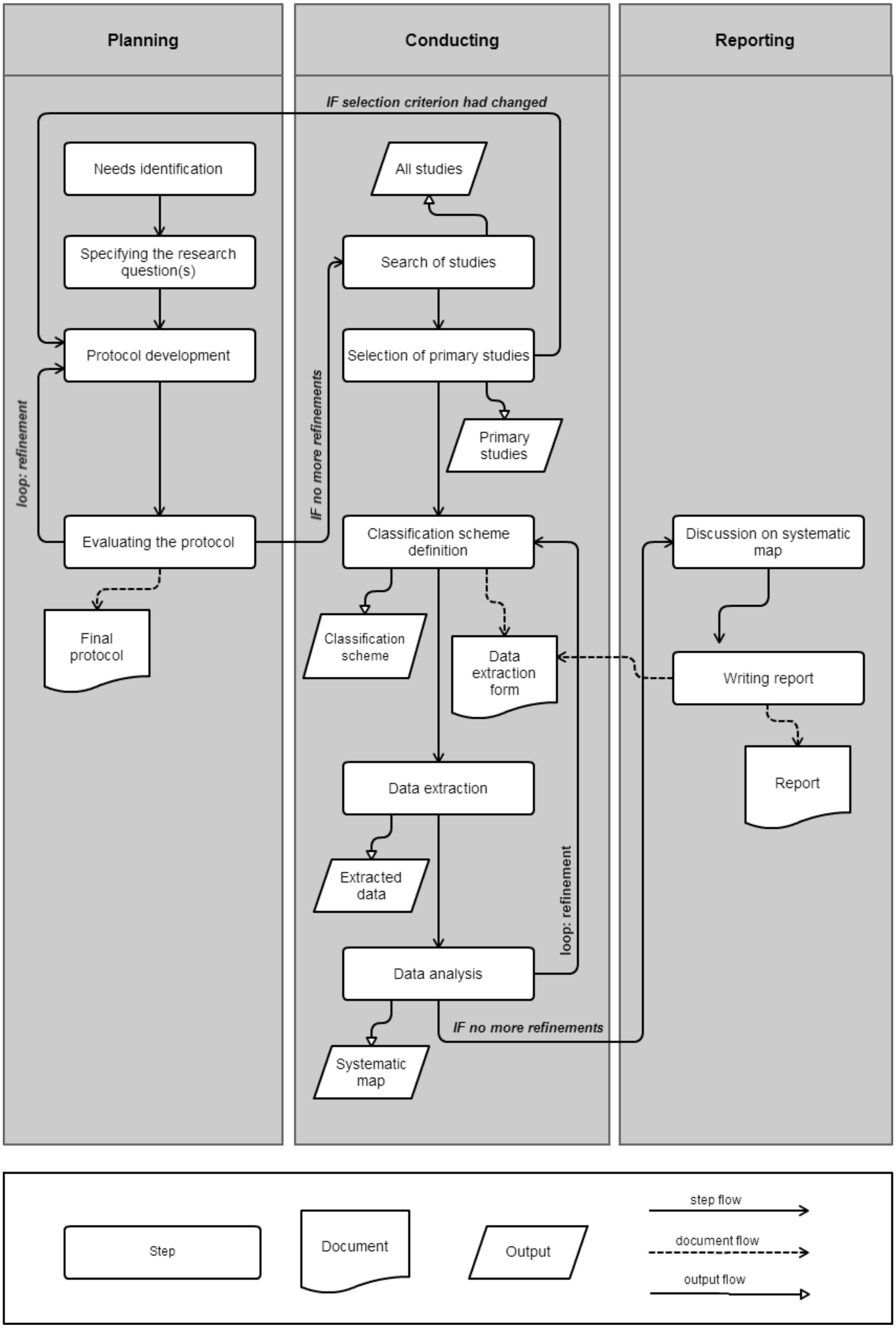

Figure 2: The Systematic Mapping Process

3.2 Team

The team is composed of several people organized as follows:

Principal Researchers: Giuseppina Casalaro, Giulio Cattivera
Master students in Software Engineering with knowledge in MDE and MMRSs. They perform most of the tasks of the systematic mapping study.

Secondary Researchers: Ivano Malavolta, Federico Ciccozzi, Andreas Wortmann
Researchers active in the field of robotics and MDE, with knowledge also in systematic literature reviews and systematic mapping studies. They constantly support and review the work carried out by the principal researchers. They also resolve conflicts to 'avoid endless discussions' among principal researchers [6].

Advisor: Patrizio Pelliccione
Senior researcher with many-years expertise in Software Engineering. He supports researchers during the data analysis and findings analysis.

Both the principal researchers together with Federico Ciccozzi (secondary researcher) are at Mälardalen University in Västerås, Sweden. The advisor Patrizio Pelliccione is at Chalmers University in Gothenburg, Sweden while Ivano Malavolta (secondary researcher) is located at the University of L'Aquila, Italy.

4. Research questions

Considering that the purpose of this systematic map is to show the current state-of-the-art for MDE applied to MMRSs, then we have identified the following overall research question: **What is the state-of-the-art of MDE for MMRSs?**
Then, in order to address this research question we have first applied the PICOC criterion in accordance with [5], and secondly we have also defined a set of sub-research questions that address different aspects of the state-of-the-art for MDE applied to MMRSs.

Our PICOC criterion definition is the following:

- POPULATION: Mobile multi-robot systems
- INTERVENTION: Model-driven engineering approaches for mobile multi-robot systems
- COMPARISON: not applicable
- OUTCOMES: a classification of the primary studies that reflects the current state of the art of MDE for MMRSs.
- CONTEXT: academic peer-reviewed publications with a software engineering perspective.

Then, the sub-research questions are:

RQ1: What are publication trends of the studies about Model-Driven Engineering for Mobile Multi-Robot Systems?

Objective: existing research on MDE for MMRSs. This research question is divided into specific sub-questions that will be addressed.

- **RQ1.1:** Which are the most target venues that address the topic of Model-Driven Engineering for Mobile Multi-Robot Systems?
- **RQ1.2:** What types of research have been applied and to what extent? (validation research, evaluation research, solution proposal, philosophical papers, opinion papers, experience papers)
- **RQ1.3**: What are the contribution types?
- **RQ1.4**: What is the publication rate by year?

Outcomes: The outcome of this research question will illustrate the current state of publication on the topic. A map that classifies the collected primary studies according to publication year, venue, author, contribution type and research type.

RQ2: Which kind of MMRSs are supported by existing MDE methods and techniques?

Objective: to identify and classify the type of MMRSs that have been engineered by MDE methods and techniques.

Outcomes: This research question will illustrate characteristics of MMRSs supported by MDE methodology, like: types of robots (e.g. aquatic, flying, terrestrial, etc.), number of involved robots (e.g. tens, thousands, etc.), movement types (e.g. vision, GPS, etc.)

RQ3: How do existing MDE methods and techniques support MMRSs?

‘Objective: To classify the current MDE methods and techniques for MMRSs by showing the adopted MDE-based strategies and the aspect(s) of the robot that is engineered. This research question is investigated through the following subquestions.

- **RQ3.1:** In which way MDE methods and techniques support MMRSs?

Outcomes: This research question will illustrate the way MDE methods and techniques support MMRSs. The scenario will be illustrated by surveying on the following categories [7]:

(1) Domain-specific modeling languages

(2) Automation: engines, transformations and generators

- **RQ3.2:** Which aspect(s) of MMRSs is engineered by MDE methods and techniques?

Outcomes: This research question will illustrate aspect(s) of MMRSs that are engineered by MDE methods and techniques (e.g. behavior, movement, image recognition, path finding, whole system, etc.).

RQ4: Which type of validation has been done for the MDE methods/techniques that support MMRSs?

Objective: To classify the type of validation or evaluation of MDE methods/techniques for MMRSs according to [8]

Outcomes: A map that will illustrate in which way MDE methods or techniques for MMRSs have been validated or evaluated. For example: real-world software projects, empirical examples, case study, experiment, formal analysis.

N.B. this research question focuses on the type of validation that a primary study is providing to support the validity of its proposed method or technique. While RQ1.2 focuses on the type of research that a primary study is showing.

RQ5: Which tools have been proposed for realizing MDE methods and techniques for MMRSs?

Objective: To identify the tools achieving MDE methods/techniques for MMRSs.

Outcomes: This research question will illustrate the MDE based tools that are used to realize MMRSs, showing the type (e.g. eclipse-based, proprietary etc.).

5. Search and selection strategy

In a secondary research, the search and selection strategy of primary papers is a critical phase that ensures a comprehensive coverage of the topic under consideration. For this reason it must be strictly defined and documented. An effective search and selection strategy should include many relevant papers but in the same time it should also exclude many of the irrelevant ones. There are two measures that are generally used for describing the accuracy of a search strategy: precision and recall. Recall is defined as the proportion between retrieved relevant material and all available relevant material, while precision is defined as the proportion of relevant material between the retrieved material. However it's unlikely to define a search strategy that can bring 100% of both precision and recall, thus, we should come out with a good trade-off resulting in a manageable volume of irrelevant material included and not too much relevant material missed [15].

Our search and selection strategy is divided in 4 steps. First, we carry out an automatic search on the most relevant scientific digital libraries, then we apply some criteria on the gathered papers that allow us to include only papers that are pertinent to the topic. Then we continue with augmenting this set through the manual technique of snowballing and eventually we eliminate all duplicates. Figure 2 shows the composition of our search and selection strategy.

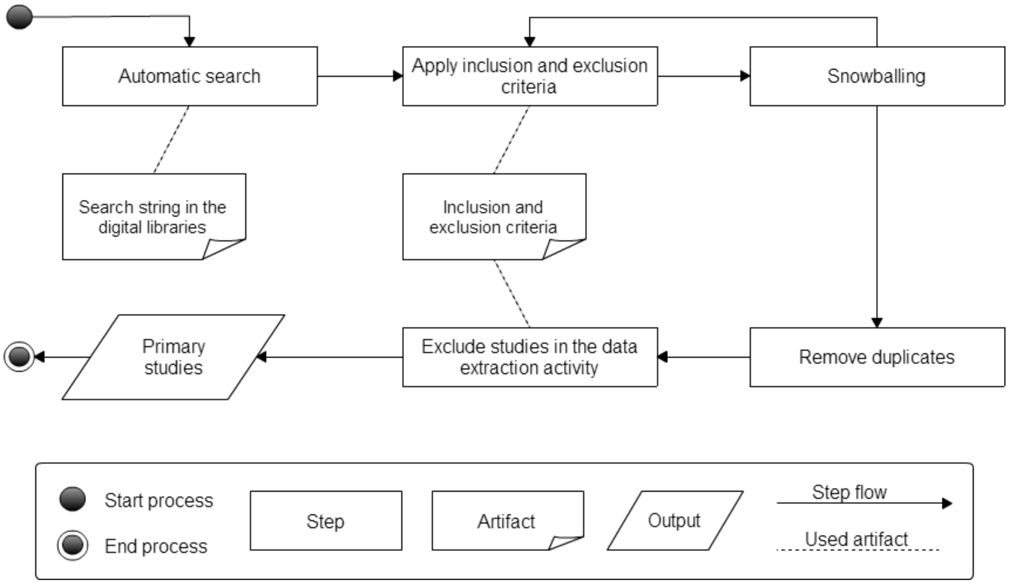
Automatic search
Apply inclusion and exclusion criteria
Snowballing
Search string in the digital libraries
Inclusion and exclusion criteria
Primary studies
Exclude studies in the data extraction activity
Remove duplicates
Start process
End process
Step
Artifact
Output
Step flow
Used artifact

Figure 3: Search and Selection Strategy

Step 1: Automatic search

In order to identify as much as possible pertinent papers, we chose to perform automatic searches on many digital libraries since it is the most popular method used in systematic studies [5]. These automatic searches have been carried out on the *{add the date}*.

Electronic Libraries

We chose ones of most important digital libraries for scientific literature that have been recognized as high valuable sources of papers in Software Engineering [17, 18, 22]: *ACM Digital Library, IEEE Xplore Digital Library, ScienceDirect, Scopus, Web of Science.* In accordance with [5] we have included at least two indexers (Scopus and Web of Science) together with real databases (all the others). Moreover these digital libraries support functionalities that have been proven to be very effective for performing automatic searches: (i) advanced command search that permits to perform complex query strings and (ii) exporting of the search results to well-defined formats like CSV, in order to perform data manipulation on the results.

Digital Library	**URL**
ACM Digital Library	http://dl.acm.org/
IEEE Xplore Digital Library	http://ieeexplore.ieee.org/
ScienceDirect	http://www.sciencedirect.com/
Scopus	http://www.scopus.com/
Web of Science	http://www.webofknowledge.com/

Table 1: Digital Libraries used for the automatic research

Search string

The search string is created based on PICOC [5]

POPULATION: Mobile multi-robot systems	(mobile OR driv* OR cruis* OR rover OR ground OR *water* OR aer* OR fly* OR sail*) AND (robot* OR vehicle* OR unmanned OR self OR autonomous)
INTERVENTION: Model driven engineering approaches for mobile multi-robot systems	(MDE OR MDD OR MDA OR MDSD OR meta-model OR metamodel OR dsl OR domain-specific OR dsml OR model-driven OR model driven)
COMPARISON	Not applicable

OUTCOMES	Classification scheme
CONTEXT	Academic peer-reviewed publications in the context of MDE applied to MMRSs

Table 2: The PICOC structure of the search string

Syntax of the search string:

(mobile OR driv* OR cruis* OR rover OR ground OR *water* OR aer* OR fly* OR sail*) **AND**

(unmanned OR self OR autonomous OR robot* OR vehicle*) **AND**

(MDE OR MDD OR MDA OR MDSD OR meta-model OR metamodel OR dsl OR domain-specific OR dsml OR model-driven OR model driven)

The search strings used in practice can be slightly different, depending on the syntax of each digital library.

Step 2: Eliminating the exact duplicates and combining equal studies

First of all, we insert all the studies in a spreadsheet.

- We consider two or more papers to be exactly duplicated if the followings match: title, authors, publication year and the venue of publication. In case of different versions of the same paper, the most recent version will be chosen.
- In case a study is published in different type of papers (e.g. both in a conference and in a journal paper) we count only one instance of the study, according to the completeness of the publication. Mostly the journal version will be kept since it is generally the most complete, but in accordance to [16] when it comes to the data extraction phase also the others will be considered.

Step 3: Selection of primary studies

In this step we apply the inclusion and exclusion criteria. These criteria permits us to consider only those papers that are pertinent to the research questions and to discard all the others. In a first round of selection, these criteria are applied considering the title of the potentially relevant study; then, if a decision cannot be made based on the title only (e.g., in case of studies that are obviously out of scope), we progressively look into more details of the study by considering also (i) the abstract and keywords and (ii) introduction and conclusions. Finally, if a solid decision about the selection of the study cannot be taken yet, the full text of the study is inspected[1]. Inclusion and exclusion criteria will be discussed in detail in Section 6.

Step 4: Forward snowballing

The starting set of papers will be augmented with the procedure of "forward snowballing", in accordance to [11]. This procedure consists of identifying those papers citing the papers being examined (primary studies), they will be found using the "cited by" feature of Google Scholar. The iterations loop ends when no new papers are found by using forward snowballing.

Step 5: Combining the primary studies

Once the gathering of papers is completed we proceed in grouping them together and eliminating duplicates by doing the following activities, like in Step 2:

1) eliminating the exact duplicates that can be possibly generated from the forward snowballing activity.
2) collapsing equal studies.

6. Selection criteria

The use of these criteria aims to identify the primary studies that completely match the purposes of our systematic map. The criteria are applied to the papers collected during the automatic search and snowballing phase. In order to reduce the likelihood of bias, these criteria are determined during the protocol definition and could be refined during the search process [5]. These criteria are composed of inclusion and exclusion criteria and are based on the research questions. They are applied to Title, Abstract, Keywords, Introduction and Conclusion sections of a study, and, if needed, to the full text of the study.

Considering a paper, it is part of the primary studies if it meets all the inclusion criteria and none of the exclusion criteria. On the other hand, a paper is not part of the primary studies if it meets at least one of the exclusion criteria.

6.1 Inclusion criteria

I1) Studies proposing MDE methods or techniques for MMRSs. We consider the definition of methodology given by [20] where it is considered as a set of methods and techniques, possibly supported by a tool.

[1] It is important to note that originally our protocol included only one step in which Title, Abstract, and keywords were checked, before analyzing (i) introduction and conclusions and then (ii) the full text of a potentially-relevant study. However, after piloting the selection we decided to check only the title in the first check of a potentially-relevant study since we noticed that there were many occurrences of studies which were clearly out of scope (even just by reading their title), so we decided to keep this first filtering as lightweight as possible (thus focussing only on the title).

I2) Studies focussing on MDE for MMRSs analyzed from a Software Engineering perspective. For example we do not consider studies that cite MDE as a background and that actually focus on robotics engineering, mechanical or industrial, hardware etc.

I3) Studies including a kind of evaluation or validation of the proposed MDE method or technique. For example: real-world software projects, empirical examples, case study, survey, experiment, formal analysis.

I4) Peer-reviewed studies: journal papers, workshop papers, conference papers.

I5) Studies written in English language.

I6) Studies available in full-text.

6.2 Exclusion criteria

E1) Secondary literature studies.

E2) Tutorial papers, short papers, poster papers, editorials, books, keynotes, tutorial summaries, tool demonstrations and panel discussions, books, introductory papers for books and workshops, technical reports and other non peer-reviewed publications.

E3) Studies not in software engineering field (e.g. robotics engineering, mechanical or industrial, hardware etc.).

E4) Studies published before 2000. Since MDE is generally recognized to be born after the OMG standard for model-driven architecture proposed in 2000 [23].

In order to reduce possible biases, three researchers will be involved in this phase of the study. Firstly, three researchers will pilot the selection procedure by independently assessing a random sample of ten potentially relevant studies. Based on the results of the pilot selection, we will either refine the selection criteria or two researchers will proceed with the independent assessment of all the other potentially relevant studies. The inter-researcher agreement using the Cohen Kappa statistic will be reported as a quality assessment of this stage of our study. Finally, all conflicts will be resolved by discussing them with a third researcher, who will play the role of arbiter so as to avoid endless discussions.

7. Classification framework definition

These are the **facets** that pertain to our study. The classification scheme will be composed of these facets, each of them addressing a research question:

1- Publication and research trends (RQ1)
2- Characteristics of MMRSs (RQ2)
3- Characteristics of MDE methods and techniques (RQ3)
4- Evaluation and validation of methods and techniques (RQ4)
5- Type of tools (RQ5)

7.1 Publication and research trends

The following are the type of data that we'll extract from the papers in order to show publication and research trends of MDE in MMRs.

- *Year of publication:* to obtain the publication tendency per year.
- *Publication venue:* to obtain their distribution and the distribution over time
  - conference papers
  - journal papers
  - workshop papers
  - book chapters
- *Research group:* to obtain (i) the number of authors that have been worked on the topic, and (ii) for each author, how many of the collected primary studies he/she has contributed for.
- *Type of research:* we refer to the classification given in [8], since it is used mainly for systematic mapping studies rather than for literature reviews. A paper can be classified in one or many of the following categories:
  - validation research
  - evaluation research
  - solution proposal
  - philosophical papers
  - opinion papers
  - experience papers

Category	Description
Validation Research	Investigated techniques 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 are the consequences of the implementation in terms of benefits and drawbacks (implementation evaluation). This also includes identification of problems in industrial settings.
Solution Proposal	A solution for a problem is proposed, and can be either novel or a significant extension of an existing one. The potential benefits and the applicability of the solution is shown by 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 and research methodologies.
Experience Papers	Experience papers explain on what and how something has been done in practice. It mirrors the personal experience of the author.

Table 3: Research Type Facets [4]

- *Contribution type:* this facet results from [4] and we will use it because it is widely used in the literature of systematic studies [14]. A paper can be classified in one or many of the following categories:
    - model
    - method
    - metric
    - tool
    - open items

Category	**Description**
Model	It refers to papers that conceptually discuss information, representations, and abstractions used in MDE for MMRSs.
Method	It refers to papers that present general concepts and working procedures to address specific concerns about MDE for MMRSs.

Metric	It refers to papers that focus on proposing or applying metrics to evaluate certain properties of MDE for MMRSs.
Tool	It refers to papers that focus on providing tool support for MDE for MMRSs, either in the form of a prototype or a tool that can be integrated with existing frameworks.
Open items	It refers to papers that presents identified issues and challenges that must be addressed by further research.

Table 4: Contribution Type Facets

7.2 MDE Methods/Techniques and MMRSs characteristics

The facets of RQ2, RQ3, RQ4 and RQ5 are created by using the method of "keywording" [4]. Keywording is a way to create a classification scheme when facets cannot be known a priori.

Steps of keywording:

- *Collecting:* For each research paper we read abstract, introduction and conclusion. While reading, we collect keywords and concepts that pertain to MDE techniques/methods and characteristics of MMRSs that are achievable by them. We keep in mind possible synonyms and homonyms that possibly might deceive us during this phase. Regarding RQ3 in particular, the keywording will be done by inspecting the following areas, in accordance with [7]:
    - Domain-specific modeling languages (DSLs): this area focuses on languages created specifically for a certain domain, context, or company to promote the development of robot in that domain. In other words, this area focuses on what is going to be modeled and with which concepts the modeling is carried out.
    - Automation: this area focuses on engines, transformations (like model-to-model, model-to-text etc.) and generators of executable code.. In other words the focus is on the purposes of models.

  At the end of this step, the set of keywords from different papers are combined together [4].

- *Clustering:* Once the collection of keywords from all papers is complete, all generated keywords are grouped into equivalence classes called clusters. In this way each cluster finally provides a facet. Then, since we are in the context of RQ2, RQ3, RQ4 and RQ5, each of the founded facet describes a method/technique or some characteristics of MMRSs.

The approach we followed for keywording is summarized in Figure 4:

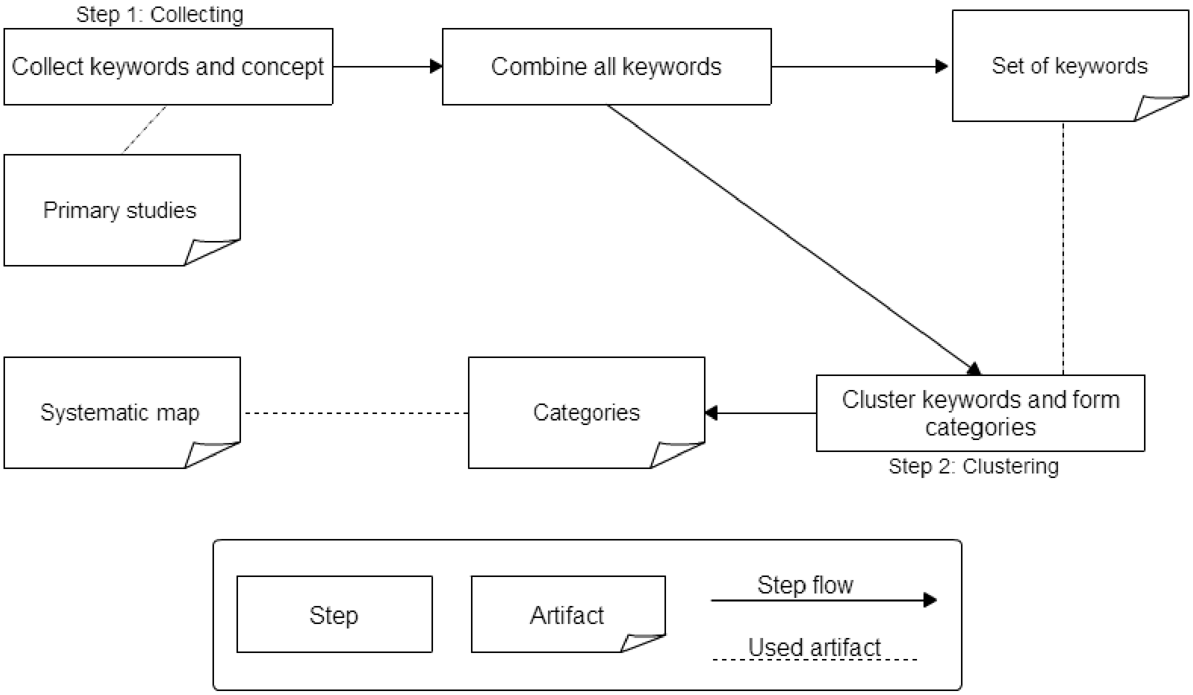

Figure 4: Keywording process

8. Data extraction

The data extraction form is a well-defined form that includes data extracted from each primary study and it is formalized by the classification scheme [4]. Formally, it is a spreadsheet in which the columns represent the attributes defined by the facets, and each row represents a study.

In order to ensure a reasonable level of agreement between the researchers during the data extraction, the following two activities will be performed together with the use of the Cohen Kappa coefficient:

Activity 1: The data extraction form will be independently filled by each of the principal researchers during the extraction of data from the set of primary studies.

The Cohen Kappa coefficient will be calculated for each attribute of a study, then for each study, the associated final value of the Cohen Kappa coefficient is the result of the average of all Cohen Kappa coefficients previously calculated for each attribute of the considered study [5].

Activity 2: An empty copy of the data extraction form will be filled by the secondary researchers by extracting data from sample studies. Sample studies are chosen randomly. Then the Cohen Kappa statistic will be applied in the same way as for Activity1.

These two activities will be performed independently by the principal and the secondary researchers to ensure bias avoidance during the data extraction process.

In case of conflict between the two principal researchers, the dispute will be solved by discussing together. After a decision has been made, then the secondary researchers will investigate the dispute and will establish if the reached solution is reasonable.

At the end of the data extraction activity, the Cohen Kappa statistic for each study must be greater or equal to 0.80 [5], otherwise each disagreement must be discussed and resolved by principal and secondary researchers.

9. Data analysis

This phase involves analyzing, counting, clustering, summarizing and classifying data extracted from the primary studies in order to produce the actual systematic map about MDE methods and techniques for MMRSs. Clearly, data analysis provides answers for the research questions defined in Section 4.

We will perform both qualitative and quantitative analysis depending on the research questions. All research questions will be addressed with quantitative analysis, and RQ2, RQ3, RQ4 and RQ5 will be addressed also with qualitative analysis.

Regarding the quantitative analysis method, it consists of two phases:

- *counting:* counting the number of primary studies for each category of the classification framework (see Section 7).
- *mapping:* arranging and displaying data from the extraction forms in a consistent and readable way with respect to the research questions (bubble plots, tables, bar charts, cake charts, etc.).

Regarding the qualitative analysis, we will perform *vertical* and *horizontal analysis* in a way according to [21]:

- Vertical analysis:
  For each research question we consider the related attributes in the extraction form, then for each of them, we analyze the data that come from each paper. The aim is to extract the main trends about the topics associated to the research questions.

For *closed-end parameters* in order to extract trends we consider the occurrence of each possible answer.

For *open-end questions* we identify categories whose definition comes out both from a keywording performed on the answers and from the relevant concepts of MDE. Then the observations and reasonings of the analysis are carried out from them.
The coding method will be also performed in support of the open-end questions analysis. It aims at encoding qualitative data coming out from the questions in order to apply quantitative and statistical data analyses techniques. Both the categorization and the coding phases will be supervised by the secondary researchers in order to mitigate possible biases between principal researchers. The outputs of this step are: (i) the categorization of responses; (ii) a description of the trends from the statistical analyses.

- Horizontal analysis:
  We investigate the data resulting from the answers in order to discover possible relations across related questions. Basically we cross-tabulate and group the answers, and make comparisons between two or more nominal attributes of the extraction form. The aims of our horizontal analysis are two:

  Checking consistency of the given answers by cross-analyzing different questions.
  Verifying hypotheses that we had previously done during the pursuance of the mapping study. Mainly we will use cross-tabulation for the verification of our hypotheses.

10. Threats to Validity

Threats to validity for this mapping study are classified according to common guidelines [16].

Construct validity:
Construct validity represents how the mapping study really reflects the intent of the researchers and what really is asked by the research questions.
First of all we we want to argue that we have followed the guidelines for systematic maps/reviews in Software Engineering by [4] and [5].
Both MDE and MMRSs are very well established terms and thus they are strongly stable to be used in a search string. Moreover to enhance the results we have used the snowballing sampling method, that has been proven to be very effective, especially in specific technical areas [15].
We assure also the depth of our research by investigating for primary studies in the most accessed and complete scientific fora for scientific publications [15, 17, 18, 22].

Internal validity:

Internal validity includes all those threats that can lead to a causal relationship between the treatment and the results, which does not reflect reality. Regarding the data analysis activity, we will use the descriptive statistics, then we can be reasonably confident that the threats are minimal [19]. The use of a rigorously defined protocol and the use of rigorous data extraction form mitigate this kind of biases.

External validity:

According to [16], threats to external validity are those that limit the generalizability of the work produced, ie the systematic map and the final results.

In our research the set of primary studies may not represent the entire set of existing studies on the topic (MDE for MMRSs). This threat is mitigated by the choice of our search strategy:

- First of all, our search strategy consists in both an automatic search, and then the forward snowballing. This second step allows us to include in the set of primary studies all those papers that were not captured by the search string in digital libraries. Furthermore, the forward snowballing ensures that the most recent relevant publications are included in our study.
- Secondly, our protocol provides inclusion and exclusion criteria which are used to refine the set of primary studies, including only studies that meet the topic. Moreover, we choose only studies in English because it is the language used for high-quality scientific papers. Other studies written in other languages concerning our topic may exist, but we consider this threat minimal. We choose to exclude the grey literature for the same quality motivation.

Conclusion validity:

Threats to conclusion validity are conditions that may invalidate the conclusions of the work [16]. The conclusions resulting from the relationship between the extracted and synthesized data and the map produced and findings.

In our study, there may be biases in the data extraction activity and in the classification scheme. We will mitigate these biases using:

- a rigorously defined protocol, that is reusable by other researchers for reproducing our study, and a rigorous data extraction form;
- the data extraction form will be filled independently by principal researchers and it will be validated by secondary researchers;
- a disciplined definition of the classification scheme;
- this protocol and the final report of this study will be preliminary evaluated by external independent researchers.

4. References

[1] T. Stahl, M. Voelter, K. Czarnecki. *Model-Driven Software Development: Technology, Engineering, Management*. John Wiley & Sons, 2006.

[2] R. Siegwart, I. Nourbakhsh, D.Scaramuzza. *Introduction to autonomous mobile robots*. MIT press, 2011.

[3] P. A. Baer, R. Reichle, K. Geihs. The SPICA Development Framework. *Intelligent Autonomous Systems 10: IAS-10* (2008): 211.

[4] 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

[5] B. Kitchenham. Guidelines for performing systematic literature reviews in software engineering. Software Engineering Group School of Computer Science and Mathematics Keele University and Department of Computer Science University of Durham, Tech. Rep., 2007.

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

[7] D. C. Schmidt. Guest editor's introduction: Model-driven engineering. *Computer* 39.2 (2006): 0025-31.

[8] R. Wieringa, N. Maiden, N. Mead, C. Rolland. Requirements engineering paper classification and evaluation criteria: a proposal and a discussion. *Requirements Engineering* 11, no. 1 (2006): 102-107.

[9] D. Di Ruscio, I. Malavolta, P. Pelliccione. A family of domain-specific languages for specifying civilian missions of multi-robot systems. In: First Workshop on Model-Driven Robot Software Engineering - MORSE 2014 (2014)

[10] M. Brambilla, J. Cabot, M. Wimmer. "Model-driven software engineering in practice." *Synthesis Lectures on Software Engineering* 1.1 (2012): 1-182.

[11] C. Wohlin. "Guidelines for snowballing in systematic literature studies and a replication in software engineering." *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*. ACM, 2014.

[12] A. Fernandez, E. Insfran, S. Abrahão. "Usability evaluation methods for the web: A systematic mapping study." *Information and Software Technology* 53.8 (2011): 789-817.

[13] B. A. Kitchenham, S. L. Pfleeger, L. M. Pickard, P. W. Jones, D. C. Hoaglin, K. El Emam, & J. Rosenberg, (2002). Preliminary guidelines for empirical research in software engineering. *Software Engineering, IEEE Transactions on*,*28*(8), 721-734.

[14] A. Mehmood, and D. N.A. Jawawi. "Aspect-oriented model-driven code generation: A systematic mapping study." *Information and Software Technology* 55.2 (2013): 395-411.

[15] P. Runeson, M. Skoglund, Reference-based search strategies in systematic reviews, in: 13th International Conference on Empirical Assessment & Evaluation in Software Engineering, Durham University, UK, 2009.

[16] C.Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, & A. Wesslén. (2012). *Experimentation in software engineering*. Springer Science & Business Media.

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

[18] Kitchenham, Barbara A., David Budgen, and O. Pearl Brereton. "Using mapping studies as the basis for further research–a participant-observer case study."*Information and Software Technology* 53.6 (2011): 638-651.

[19] Engström, E., & Runeson, P. (2011). Software product line testing–a systematic mapping study. *Information and Software Technology*, *53*(1), 2-13.

[20] C. Ghezzi, M. Jazayeri, D. Mandrioli, Fundamentals of Software Engineering, 2nd Edition, Prentice Hall PTR, Upper Saddle River, NJ, USA, 2002.

[21] Malavolta, I., Lago, P., Muccini, H., Pelliccione, P., & Tang, A. (2013). What industry needs from architectural languages: A survey. *Software Engineering, IEEE Transactions on*, *39*(6), 869-891.

[22] Dybå, T., Dingsøyr, T., & Hanssen, G. K. (2007, September). Applying Systematic Reviews to Diverse Study Types: An Experience Report. In *ESEM*(Vol. 7, pp. 225-234).

[23] Soley, R., the OMG Staff Strategy Group: Model Driven Architecture, OMG Document

omg/2000-11-05. (2000). Available at http://www.omg.org/cgi-bin/doc?omg/2000-11-05