This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2733518, IEEE Access

Model-Driven Reverse Engineering Approaches: A Systematic Literature Review

Claudia Raibulet, Francesca Arcelli Fontana and Marco Zanoni

***Abstract*—This paper explores and describes the state of the art for what concerns the model-driven approaches proposed in the literature to support reverse engineering. We conducted a systematic literature review on this topic with the aim to answer three research questions.**

We focus on various solutions developed for model-driven reverse engineering, outlining in particular the models they use and the transformations applied to the models. We consider also the tools used for model definition, extraction, and transformation and the level of automation reached by the available tools.

The model-driven reverse engineering approaches are also analyzed based on various features such as genericity, extensibility, automation of the reverse engineering process, and coverage of the full or partial source artifacts. We describe in detail and compare fifteen approaches applying model-driven reverse engineering. Based on this analysis, we identify and indicate some hints on choosing a model-driven reverse engineering approach from the available ones, and we outline open issues concerning the model-driven reverse engineering approaches.

***Index Terms*—Models, Reverse engineering, Model-driven reverse engineering, Model transformation, Legacy system**

I. Introduction

Recent researches reveal a paradigm shift in the software engineering field: from the object-oriented where *everything is an object*, to the models-oriented where *everything is a model* [1]. In this context, models are representations at high abstraction level of the whole system or of a part of it. The Object Management Group (OMG) [2] defines models as: "a model of a system is a *description* or *specification* of that system and its environment for some certain purpose" [3].

The use of models in software engineering is the foundation of *model-driven engineering* (MDE), i.e., the unification of initiatives that aim to improve software development by employing high-level, domain-specific models in the implementation, integration, maintenance, and testing of software systems [4], [5].

In the literature, MDE is the general term for all model-based principles and techniques that can be applied to both forward engineering and reverse engineering. More and more works appear, not always related to software engineering, focused on the use of models to alleviate the inherent complexity of given tasks. MDE approaches have been proposed in the context of forward engineering, where appropriate tools can translate high level models into source code. One of the key ideas in MDE is that transformation of models from a high abstraction level to a lower level can be described and automated by using transformation languages. So, source code can be generated with a small amount of hand written code, less effort, and possibly less errors [6]. The use of models allows developers to focus on the important aspects of the system, neglecting technical details specific to the target platform [7].

Nowadays, systems are not developed from scratch and a *reverse engineering* phase is often required. Reverse engineering can be defined as the process of comprehending software and producing a model of it at a higher abstraction level than source code, suitable for documentation, comprehension, maintenance, or reengineering [8]. Reverse engineering can be applied in many contexts. The most common scenario is the comprehension of code, e.g., maintenance, software evolution, integration or interface with legacy systems. Another scenario is the re-documentation of a system, when old documentation has to be updated after changes of the source code. Further scenarios in which reverse engineering can be exploited are the migrations of legacy systems to new platforms or to new development paradigms, e.g., the use of MDE for an existing system. Moreover, reverse engineering can be used for software quality assessment and for extracting from source code metrics or other hints on the quality of the code. Scalise et al. [9] describe how MDE can be applied to develop software comprehension tools, useful during reverse engineering activities.

The application of MDE to solve reverse engineering issues is called *Model-Driven Reverse Engineering* (MDRE). Favre [10] defines MDRE as "producing descriptive models from existing systems that were previously produced somehow".

Generally, the MDRE process is made by the following steps:

1) get a view (i.e., a model) of the analyzed legacy system from source artifacts;
2) exploit the model to achieve a specific goal, e.g., redocumenting or reengineering a system.

Often, in MDRE the only available source of knowledge is the source code, so MDRE solutions start from a system model with a low abstraction level (the source code) and try to build views at higher abstraction levels. Many different views of the system are needed, each corresponding to a different model. The models generation from source code and models transformations can be automated. After the generation, the obtained models can be analyzed by domain experts or by appropriate tools, otherwise they can be used to start a model-driven development phase.

Reverse engineering approaches and tools use models to abstract the underlying system. This is natural, considering that to represent the analyzed project in a more abstract way some

C. Raibulet, F. Arcelli Fontana and M. Zanoni are with the University of Milano-Bicocca, Viale Sarca, 336, Milano, Italy
E-mail: {raibulet,arcelli,marco.zanoni}@disco.unimib.it

kind of model is needed. What is different in MDRE is that the whole process is based on the systematic use of models to represent information, and the use of metamodels to describe these models. Moreover, the model-to-model transformation is carried out explicitly, in addition to the discovery process, e.g., with text-to-model transformation. Existing reverse engineering approaches are often very flexible w.r.t. the metamodels (e.g., using graphs [11], [12], [13]), or in the contemporary use of source code and model analysis/transformation (e.g., Moose [14] or MARPLE-DPD [15]). From this point of view, we can consider MDRE a subset of the possible approaches for reverse engineering, where the discovery process and model manipulation are disciplined by means of MDE.

In this paper, we report a systematic literature review (SLR) [16], [17], [18] of the approaches that have been proposed in the context of MDRE.

In our past research activities, we applied modeling in reverse engineering, in the context of software architecture reconstruction [19] and design pattern detection [20], [21], [22], taking in particular into account the role of design pattern decomposition in elemental structures [23], [24]. We have been also interested in collecting the different metamodels applied to reverse engineering in the literature [25]. Given our interest in this area, we would like to understand the current state of model-driven engineering applied in the context of reverse engineering, with the aim of choosing available approaches for new problems or having hints about the choices to make when developing a new approach.

Therefore, in this survey we emphasize the principal features of these identified approaches, by focusing our attention on the following aspects:

- the definition and reuse of metamodels;
- the implementation or reuse of tools for model definition, extraction, and transformation;
- the level of automation reached by the proposed tools;
- the type of source artifacts analysis applied to build models.

The paper is organized as follows. Section II reports the systematic literature review (SLR) design and execution, and defines the research questions. Section III describes the approaches selected during the review, highlighting their main features. Section IV compares the MDRE approaches according to various aspects and criteria. Section V answers the research questions using the results of the review. Section VI discusses the threats to the validity of this work. Section VII summarizes some criteria to choose a MDRE approach and outlines some limitations of the available MDRE approaches. Finally, conclusions are dealt in Section VIII.

II. Systematic Literature Review: Research Method

This survey aims to capture the state of the art concerning the approaches for MDRE. It is a systematic literature survey that follows the general guidelines proposed by Kitchenham [16], [18]. We considered also the indications outlined by Brereton et al. [17]. The review process is composed of three main phases: planning, conducting, and reporting. The details of each of these phases are described in the remaining of this section.

A. Planning the Review

We are interested in identifying the available reverse engineering approaches based on the main concepts of the model-driven paradigm. More explicitly, we are looking for approaches which discover models from legacy source artifacts, describe transformations on the models to obtain further more abstracted models, and use metamodels to drive the discovery and transformation of models.

We formulated the following research questions (RQ) to guide our survey:

- RQ1 Which metamodels are used by the model-driven reverse engineering approaches? Are they defined to solve specific problems or are they reused for more than one purpose?
- RQ2 Which tools are used by the approaches for their implementation? Do the approaches provide new tools or do they re-use existing tools?
- RQ3 What is the level of automation of the transformations defined in the MDRE approaches?

During the planning phase, we established the review protocol that includes the following tasks: selection of the search keywords, selection of the inclusion and exclusion criteria for the candidate papers, and selection of the search engines.

We considered the following main keywords to be searched: model-driven, reverse engineering, model discovery, transformation, language, and legacy.

We selected the following search engines for our survey: IEEE Explore, ACM Digital Library, Web of Science, Scopus, and Elsevier Science Direct. We think that this set of search engines includes all the high quality software engineering journals, conferences, and workshops. Table I reports the queries applied to each search engine.

B. Conducting the Review

On September 5th, 2016 we executed the queries described in Table I. The queries went under several modifications before the version reported in the table, because of the differences among the various search engines. Initially, we formulated all the queries to search for: (1) model-driven reverse engineering (including variants) keywords in the title, abstract and keywords sections of papers, and (2) legacy, discovery, transformation, language and metamodel in the entire paper. The query for the Scopus database is the closest one to the initial query (see Table I). We had to adapt the query to each searched database because the results were very few or even zero. Therefore, we report in Table I the exact query for each searched database. To summarize, our effort to define the queries has produced comparable results from all the searched databases, except Web of Science which produced a number which represents twice the results obtained by the other four databases. We preferred to spent effort to investigate all the 80 results of Web of Science generated by a more general query instead of having fewer results which might miss some

TABLE I
SYSTEMATIC LITERATURE REVIEW QUERIES

Source	Query	Results
IEEE Explore	`("Abstract": (("model-driven" OR "model driven" OR "model-based" OR "model based") AND ("reverse engineering" OR "MDRE")))`	41
ACM Digital Library	`recordAbstract:("model driven" "model-driven" "model based" "model-based") AND recordAbstract:("reverse engineering" MDRE)`	29
Web of Science	`TOPIC: (((("model-driven") OR ("model-driven") OR ("model-based") OR ("model based")) AND ("reverse engineering" OR "MDRE")))`	80
Scopus	`(TITLE-ABS-KEY (model driven) OR TITLE-ABS-KEY (model-driven) OR TITLE-ABS-KEY (model based) OR TITLE-ABS-KEY (model-based)) AND (TITLE-ABS-KEY (reverse engineering) OR TITLE-ABS-KEY (mdre)) AND ALL (legacy) AND ALL (discovery) AND ALL (transformation) AND ALL (language) AND (ALL (metamodel) OR ALL (meta model) OR ALL (meta-model))`	27
Science Direct	`TITLE-ABSTR-KEY(("model driven" OR "model based") AND ("reverse engineering" OR "MDRE"))`	28

significant papers for our research. No constraints on the time period or type of research publication have been applied. The review has involved 3 team members: an Associate Professor, an Assistant Professor, and a Post-Doc.

As already anticipated, the exact search query has been personalized for each search engine due to their specific search interface (e.g., basic or advance search mechanisms). For IEEE Explore and ACM Digital Library we had to relax the queries because the search of all the keywords provided zero results. In Web of Science, we refined the queries using categories and research areas. We *included* results in the category *"COMPUTER SCIENCE: SOFTWARE ENGINEERING, THEORY METHODS, INFORMATION SYSTEMS, INTERDISCIPLINARY APPLICATIONS, ARTIFICIAL INTELLIGENCE"*, *excluded* results in the following research areas:

- MECHANICS,
- MATHEMATICAL COMPUTATIONAL BIOLOGY,
- PHYSICS,
- METEOROLOGY ATMOSPHERIC SCIENCES,
- MATERIALS SCIENCE,
- OPTICS,
- METALLURGY METALLURGICAL ENGINEERING,
- OPERATIONS RESEARCH MANAGEMENT SCIENCE,
- ENERGY FUELS,
- TELECOMMUNICATIONS,
- BIOTECHNOLOGY APPLIED MICROBIOLOGY,
- LIFE SCIENCES BIOMEDICINE OTHER TOPICS,
- SCIENCE TECHNOLOGY OTHER TOPICS,
- BIOCHEMISTRY MOLECULAR BIOLOGY,
- INSTRUMENTS INSTRUMENTATION,
- THERMODYNAMICS,
- ENVIRONMENTAL SCIENCES ECOLOGY,
- MATHEMATICS,
- SURGERY,
- MEDICAL INFORMATICS,
- ENGINEERING,
- PSYCHOLOGY,
- EDUCATION EDUCATIONAL RESEARCH,
- AUTOMATION CONTROL SYSTEMS OR ORTHOPEDICS,

and *excluded* the following "Web of Science Categories":

- COMPUTER SCIENCE HARDWARE ARCHITECTURE,
- COMPUTER SCIENCE CYBERNETICS.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2733518, IEEE Access

TABLE II: Systematic literature review selection results

Paper	Year	Ref	Described	Cited	IEEE	ACM	WoS	Scopus	SD
Amendola and Favre	2013	[26]	●	○	○	○	○	●	○
Arevalo et al.	2016	[27]	○	●	○	○	●	●	●
F. Barbier et al.	2010	[28]	○	●	○	○	○	○	●
G. Barbier et al.	2010	[29]	●	○	○	○	○	○	●
Bellucci et al.	2012	[30]	○	●	○	●	○	○	○
Bensaber and Malki	2008	[31]	●	○	○	●	○	○	○
Bergmayr et al.	2016	[32]	○	●	○	○	○	●	○
Bergmayr et al.	2013	[33]	●	○	○	○	●	○	○
Blanco et al.	2009	[34]	●	○	○	○	○	●	○
Bouillon et al.	2005	[35]	●	○	○	○	●	○	○
Bruneliere et al.	2014	[36]	●	○	○	○	●	○	●
Cosentino et al.	2013	[37]	●	○	●	○	●	○	○
Damasevicius et al.	2012	[38]	○	●	○	○	○	●	○
Djamel et al.	2007	[39]	●	○	●	○	○	○	○
El Beggar et al.	2013	[40]	●	○	○	○	●	○	○
Favre	2008	[41]	●	○	●	○	○	○	○
Favre et al.	2009	[42]	●	○	○	○	●	○	○
Favre et al.	2012	[43]	●	○	○	○	○	●	○
Favre et al.	2014	[44]	●	○	○	○	○	●	○
Fleurey et al.	2007	[45]	●	○	○	●	●	○	○
Garzón et al.	2014	[46]	○	●	○	●	○	○	○
Garcia-Rodriguez de Guzman et al.	2007	[47]	●	○	○	○	○	●	○
Lenk et al.	2012	[48]	●	○	○	○	●	○	○
Martinez et al.	2014	[49]	●	○	●	○	○	●	○
Martinez et al.	2013	[50]	●	○	○	○	○	●	○
Martinéz Perez et al.	2013	[51]	○	●	○	○	●	○	○
Normantas and Vasilecas	2012	[52]	●	○	○	○	●	●	○
Normantas et al.	2012	[53]	●	○	○	○	○	●	○
Ovchinnikova and Asnina	2005	[54]	○	●	●	○	○	●	○
Pereira et al.	2011	[55]	●	○	○	○	○	●	○
Pérez-Castillo et al.	2011	[56]	●	○	○	○	○	●	●
Pérez-Castillo et al.	2011	[57]	○	●	○	○	○	●	○
Pérez-Castillo et al.	2011	[58]	●	○	○	○	○	●	○
Polo et al.	2007	[59]	●	○	○	○	●	○	○
Pu et al.	2008	[60]	○	●	●	○	○	○	○
Qiao et al.	2003	[61]	○	●	○	○	●	○	○
Ristic et al.	2015	[62]	●	○	○	○	●	○	○
Ristic et al.	2014	[63]	●	○	○	○	●	●	○
Rodriguez-Echeverria et al.	2012	[64]	●	○	○	○	●	○	○
Rugaber and Stirewalt	2004	[8]	○	●	○	○	●	○	○
Sánchez Ramón et al.	2010	[65]	●	○	○	●	●	○	○
Sánchez Ramón et al.	2011	[66]	●	○	●	○	○	●	○
Sánchez Ramón et al.	2013	[67]	●	○	○	○	●	○	○
Sosa Sánchez et al.	2014	[68]	●	○	○	○	●	○	○
Sun et al.	2009	[69]	●	○	○	○	●	○	○
Trias et al.	2013	[70]	●	○	○	○	○	●	○
Trias et al.	2013	[71]	●	○	○	○	○	●	○
Trias et al.	2013	[72]	●	○	○	○	○	●	○
Trias et al.	2015	[73]	●	○	○	○	●	●	●
Trias et al.	2015	[74]	●	○	○	○	●	●	○
Warwas and Klusch	2011	[75]	●	○	○	●	○	○	○

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2733518, IEEE Access

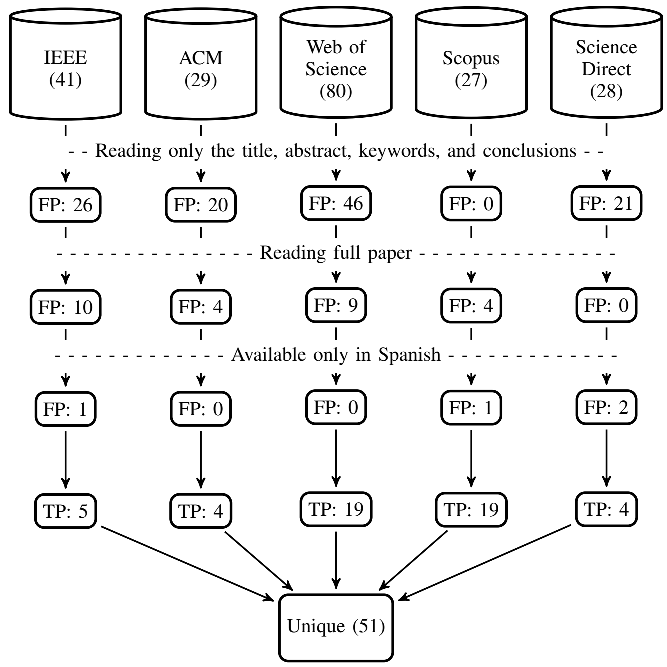

Fig. 1. Systematic literature review selection process

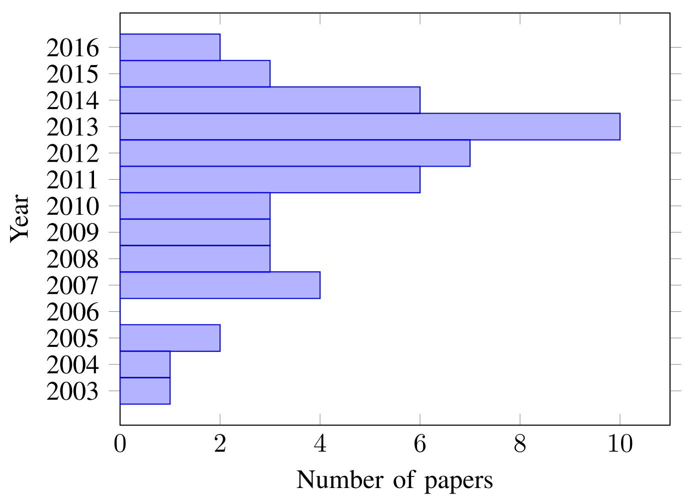

Fig. 2. Distribution of SLR TP results for each year

Figure 1 summarizes the review process and its quantitative results: the source search engine, the number of the identified papers, the number of false positives (FP) (i.e., papers which contain the keywords but do not describe a model-driven reverse engineering approach), the number of true positives (TP) (i.e., papers which describe a model-driven reverse engineering approach). The process of excluding FPs was done by reading first (for all the 205 results) the title, abstract, keywords, and conclusions, and in case of any doubt, the entire paper. All the results have been read by two of the team members (the Assistant Professor and the Post-Doc). In case of disagreements, the papers were read also by the Associate Professor, who evaluate the paper without knowing the evaluation of the other two team members. Finally, all the members met and discuss the evaluation of each paper. The FPs we excluded were mainly papers addressing model-driven engineering issues, which cite also the reverse engineering keywords without focusing on them. Furthermore, we have decided to divide the TPs into two groups: a first group containing all the approaches which are completely described by their authors by providing all the details concerning the model-driven reverse engineering process, and a second group containing the approaches which are partially described by their authors in the published papers (e.g., models and meta-models and/or transformations not completely described). The first group of approaches is presented in Section III, while the second in Appendix A. Papers not written in English have been excluded (we found some results written in Spanish).

All the papers describing a model-driven reverse engineering approach and presented in Section III have been entirely read by the team members. We have also verified that the references cited into the papers we read did not point to any other MDRE approach.

As it can be observed from Figure 1, the results obtained from the search queries are relatively low, even if we relaxed the queries for four of the five search engines. For three of the search engines, the number of the results is quite the same (i.e., ACM, Scopus, Science Direct). Also the results for IEEE are close. Instead, Web of Science provided more than double results with respect to the other search engines. However, the TPs of Web of Science are comparable to those of Scopus. The TPs identified are quite few. This may be due to the fact that the model-driven reverse engineering field is complex and it requires a significant effort for the development of approaches. In the same time, the field is young (see Figure 2).

C. Reporting the Review

To answer (in Section V) the research queries previously formulated in this paper, we provide, in Section III, a brief description of each of the identified model-driven reverse engineering approaches. The brief description consists in: an overview of the approach indicating its objectives and its application, if any (e.g., case study), a description of the models and metamodels defined or used by the approach, the reverse engineering steps implemented by the approach, and further considerations, indicating particularities of each approach not captured in the previous sections of the description. Furthermore, in Section IV we relate our findings with different features of MDRE approaches defined in the literature, while in Section VII we provide hints on choosing a MDRE approach from the available ones and outline some open issues of the existing approaches.

Table II reports the list of all the MDRE approaches produced by the selection process and from which sources they have been found. We also provide the full list of retrieved papers and their classification at http://essere.disco.unimib.it/wiki/research/mdre_slr_tr. We report in Figure 2 the distribution of the found papers over the years. From the figure, we can observe that, after an initial phase in years 2003–2006, this research topic had a constant number of publications, with no gaps in between, and reaching a peak in popularity

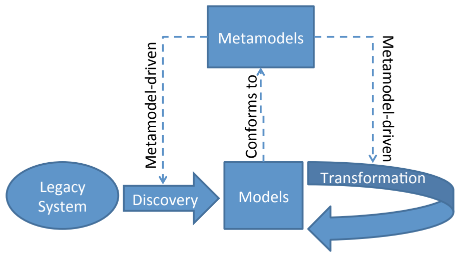

Fig. 3. MDRE process concepts

in 2013. From all the true positives papers, we introduce in Section III only the approaches that are completely described by their authors (see Table II - *Described* column), i.e., the authors provide information about legacy source code, models extracted, transformation of models, metamodels.

In Appendix A, we briefly present the MDRE approaches which are not completely described by their authors (i.e., metamodel missing information, transformations not completely described), and hence not considered in Section III. These approaches are indicated as *Cited* in Table II.

III. MDRE Approaches

In this section, we analyze the model-driven reverse engineering (MDRE) approaches we have identified as primary studies during the literature review. The approaches we consider can be mapped on the main concepts described in Figure 3. They implement discovering mechanisms through which models are obtained automatically from the analyzed legacy systems. These models have a direct correspondence with the source artifacts. The OMG standard calls these models as Platform Specific Models (PSM) [3]. Bruneliere et al. defines these models as initial models [36]. Further, these models are modified and abstracted through model transformations with the objective to obtain the target representation of the analyzed systems. Model transformations include model navigation, querying, computation, and building further models [36]. The OMG standard calls these models as Platform Independent Models (PIM) [3]. Bruneliere et al. introduces the term derived models [36], to characterize the models obtained through model-to-model transformations. All these main steps are metamodel driven. Figure 3 can be considered an abstraction of the diagrams proposed by Bruneliere et al. [36]. As Favre observes in [10], the MDRE is a complex task because “various steps are required, with one or more models being produced at each step”. Bruneliere et al. indicates the four main steps of a MDRE process:

- the exploration of the legacy system via its initial models;
- the identification of the required information via the initial models;
- the computation of views using the identified information as source;
- the representation retrieval into derived models.

To summarize, in this section, we present the MDRE approaches which describe explicitly at least one PSM or initial model and one PIM or derived model, as well as at least one text-to-model and one model-to-model transformation.

With the aim to provide a preliminary common comparison framework of the described MDRE approaches, we consider the following features:

- the *Models* used;
- the *Scope* of the approach, general purpose or specific;
- *Tool/s* used to implement the approach;
- the availability of *Case Studies*, where the approach has been applied;
- *Automation level*, if the approach is automatic or semi-automatic;
- *Type of Analysis* (static, dynamic).

The comparison of the approaches according to this framework is discussed in the next section, after the description of each approach.

Furthermore, in the literature, Bruneliere et al. [36] proposed five general characteristics a MDRE approach should have, that we partially covered through the above features of the framework:

- Genericity: A MDRE approach should be based on technology-independent standards (i.e., metamodels) and customizable model-based components; this characteristic is captured in the *Models* and *Scope* features;
- Extensibility: A MDRE approach should rely on a decoupling of the represented information (models) and the next steps of the process, i.e., belonging to MDE; in the description of the approaches we have indicated which approaches are part of a round-trip engineering solution;
- Automation: if the reverse engineering process can be totally or partially automated, the *Automation level* feature we considered in our framework;
- Full/Partial Coverage: this characteristic regards the source artifacts including interrelated views at different abstraction levels; it is captured by the *Scope* features;
- Direct (re)use and integration: The various elements of a MDRE approach and the obtained results (i.e., models) should be designed for reuse. We discuss the reuse level from the *Models*, *Tools*, and *Case Studies* features.

We discuss these characteristics in Section V, according also to the features considered for our comparison framework.

In the following, we first provide an overview of the standardized metamodels that are used in MDRE, and then we describe the selected MDRE approaches. For each of the described approaches, we provide a short overview, the models they use, the main steps of the model-driven reverse engineering process by focusing on the transformations on models, the tools used and the level of automation of the approach, and further considerations in particular related to the features we have considered for the comparison framework. The approaches are ordered alphabetically based on the name of the first author.

A. Standardized models for MDRE

The Object Management Group (OMG), through the Architecture-Driven Modernization (ADM) [76], [77] initiative, defined different models with the goal of supporting reverse engineering activities. The first metamodel they introduced is the Knowledge Discovery Metamodel (KDM) [78],

[57]. The goal of KDM is to define a shared and complete representation, able to guarantee the interoperability of different tools, and to efficiently support maintenance, evolution, assessment, and modernization activities. The model is defined at a level of detail able to represent the structural concepts of object-oriented and imperative languages, e.g., classes, methods, functions, modules. It also contains an optional package (MicroKDM) able to specify finer details about the intra-function behaviour of software, to support data and control flow analysis. KDM is structured in packages organized in different layers, where only the core layers are mandatory, and the higher are optional. The higher layers provide support to model different aspects of software, like design constructs, deployment environment, or user interface.

To better support source code analysis activities, ADM also defined the Abstract Syntax Tree Metamodel (ASTM) [79] metamodel, to represent the Abstract Syntax Tree (AST) of virtually any programming language, allowing analysis tools to target the metamodel instead of the specific language's AST. The model defines what is called GASTM (Generic ASTM), i.e., definitions that recurrently apply to ASTs of most programming languages, and allow extensions, called SASTM (Specialized ASTM), to handle features specific of a single programming language.

More recently, other metamodels have been introduced to support the representation of software metrics (or any kind of metric) applied to existing models (SMM [80]), or the representation of software patterns (SPMS [81]). Finally, the Consortium for IT Software Quality (CISQ)[1] contributed to the definition of metamodels integrated with the ADM standards, adding the representation of different quality measures, e.g., maintainability (ASCMM [82]).

All the models defined by ADM use the OMG metamodeling stack, which includes the MOF [83] meta-meta-model, and the XMI serialization format. Since Eclipse EMF[2] is compatible with MOF (ECore is compatible with EMOF), these models can be imported and applied using EMF technologies. Anyway, the specifications are available, so any other kind of implementation can be provided in theory for the metamodels.

B. Bruneliere et al.

Bruneliere et al. [29], [36] propose MoDisco (Model Discovery), an Eclipse open source project for model-driven reverse engineering of IT systems. Its main objective is to provide support for activities dealing with legacy systems and ranging from understanding and documentation to evolution, modernization, and quality assurance.

MoDisco has a modular architecture based on three layers: infrastructure, technologies, and use cases. The infrastructure layer offers the generic reusable components used in the MDRE process (e.g., generic metamodels, model transformation, model navigation, model customization, model orchestration). The technology layer provides coverage for legacy technologies (e.g., technology-dedicated components independent from any reverse engineering scenario). The use case scenarios introduce reuse and integration examples.

It is worth noting that MoDisco is cited by the ADM Task Force of OMG as an implementation example of its standards KDM, Structured Metrics Metamodel (SMM), and ASTM.

1) MDRE Models: MoDisco is a generic and extensible metamodel-driven approach to model discovery, understanding, and transformation. It provides a core metamodel approach based on the OMG KDM specification. Furthermore, it enables the extension of the metamodel for different domains in various fields. A metamodel can be generic or technology specific, depending on the current reverse engineering needs. For example, MoDisco provides a complete metamodel for Java, JSP, and XML.

Models are obtained through software components called discoverers. Transformations or chains of transformations may be applied to the extracted models to obtain the desired view of the investigated system.

2) MDRE Steps: The MoDisco approach consists in two main steps:

- model discovery;
- model understanding.

The model discovery step depends on the source technology of the analyzed software system. It is performed through model discoverers, which can be fully hard-coded or partially generated. Model discoverers may imply two types of operations: injection and transformation. Through injection the legacy system data is extracted and represented as initial models. Transformations concern additional syntactic or structural mapping to complete the initial models.

The understanding step analyzes the initial model(s) and generate the derived models. Transformations may be applied either to obtain the desired view on the system or the desired goal of the reverse engineering process. Transformations can be partially or entirely automatized. The following sub-steps are executed in an iterative process: model navigation (e.g., exploration of the initial models), model querying (e.g., identification of required information), model computation (e.g., view computation of the query results), and model building (e.g., representation retrieval into derived models).

3) Tool Support: MoDisco provides a generic framework and several generic tools able to automatically extract models from legacy systems, models which can be further manipulated through transformations. MoDisco also offers ready-to-use support for three different legacy technologies: Java, JEE (including JSP), and XML.

4) Further Considerations: MoDisco may be considered the MDRE approach which is:

- well-motivated through several various examples;
- standardized because it is the MDRE solution which implements most of the OMG standards specification for reverse engineering and because it is an Eclipse framework (Eclipse being a de facto standard for software development);
- applied in some industrial case studies;
- generic and extensible because it defines the core components which can be applied in various domains and application scenarios.

[1] http://it-cisq.org/
[2] http://www.eclipse.org/modeling/emf/

C. Cosentino et al.

The approach proposed by Cosentino el al [84] aims to extract business rules out of Java source code by isolating the code segments concerning the business processes. To achieve this goal, the approach exploits model-driven reverse engineering concepts implementing the following main steps: model discovery from Java code, variable classification and domain variables model creation, and business rules model extraction. The model discovery step is implemented with MoDisco, while the following steps represent a model-to-model transformation chain implemented using the Atlas Transformation Language (ATL).

The approach has been validated through an application which simulates the behavior of animals and humans in a meadow, where actors (animals and humans) can act and move according to their nature.

1) MDRE Models: From the Java source code the first Java model is created. The Java model has a one-to-one correspondence with the source code and it represents the PSM model. For each domain, a domain variable model is created by identifying a domain variable and its containing classes starting from the PSM.

From the PSM and the set of the domain variable models, two further models are generated: *(1)* a business rule model containing the internal representation of the business rules belonging to each domain variable, and *(2)* a global domain model joining all the classes, method signatures, and class attributes relevant for the union of domain variable models. Finally, the business rule representation takes in input the business rule model for a domain variable and, optionally, the domain model and provides human-understandable artifacts describing the extracted business rules for a given variable.

The results can be provided in a textual form or as a generic or specific graph (conforming to the Portolan metamodel which addresses the gap between the domain data and its graph visualization).

The approach defines a variable classification metamodel and a business rule metamodel. Furthermore, the approach provides traceability support indicating also a traceability metamodel. Through traceability, the approach maintains the link between the extracted business rules and the part of the source code that justifies their extraction.

2) MDRE Steps: This approach implements the following steps:

- model discovery, exploiting the MoDisco tool, through which the PSM is created;
- variable classification, identifying only those variables concerning the application domain;
- business rule identification, composed of three sub-steps: domain model extraction, slicing operation, and business rule model extraction;
- business rule representation, providing human-understandable artifacts describing the extracted business rules.

3) Tool Support: Cosentino et al. use existing tools to implement their automatic approach: MoDisco for the discovery phase and ATL based tools for model-to-model transformations.

4) Further Considerations: Cosentino et al. propose a fully automatic approach. However, it allows user intervention at the end of each step to refine and improve the results of the extraction heuristics.

From the reusability point of view, a domain model can be reused by other software in the same application domain. The business rules extracted for a software may be exploited as auxiliary information for another software in the same domain.

Currently, the approach considers Java software, while the authors plan to extend it to other programming languages (e.g., COBOL [37]).

D. Djamel et al.

Semantic Web aims to provide support for the automation of the discovery, invocation, composition, and integration of Web services. The approach proposed by Djamel et al. [39] allows to build semantic description in OWL-S terms of Web services originally described in WSDL (Web Services Description Language). This approach aims to facilitate the degree of automation in the semantic association process for Web services through a model-driven reverse engineering solution which proposes a UML profile as an intermediary level between WSDL and OWL-S descriptions of Web services.

The OWL-S description of Web services is obtained through a forward engineering process using XSLT transformations which takes in input the UML models. In this way, a semantic description of the Web service, that can be processed by a reasoning tool, is obtained.

The approach has been validated through the CongoBuy example, while the correctness of the proposed transformation through the Protege ontology specification tool.

1) MDRE Models: Djamel et al. propose a UML profile for semantic Web Services as an extension of the UML 2.0 activity models. Another extension of the UML 2.0 activity model concerns the semantic aspect, which comes from the UML ontology profile [85], [86]. Stereotypes, tagged values, and data types of the UML profile are used to mark-up the PIM (obtained through a reverse engineering process and expressed in UML terms) to enable the transformation of the UML models into OWL-S terms (a forward engineering step out of the scope of this paper). Two UML diagrams are created for each Web service: a class diagram, to describe the interface of a service together with its operations, and an activity diagram, to model the internal behavior of the Web service operations and the order in which they are performed.

2) MDRE Steps: The reverse engineering process proposed in this approach is composed of the following two steps:

- discovery of the UML profile entities from the WSDL description of a Web service and generation of the class and activity diagrams;
- annotation of the obtained diagrams with further information such as category, pre-conditions, post-conditions, and effects.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2733518, IEEE Access

3) Tool Support: Djamel et al. use existing tools to implement their approach: a UML tool to generate the class and activity diagrams, a OWL tool to manage OWL-S ontologies and to map UML to OWL-S, and Protege ontology specification tool to validate the correctness of the transformations.

4) Further Considerations: Djamel et al. propose a semi-automatic approach for reverse engineering Web services. It is part of a round-trip solution. From the reusability point of view, it can be applied to various semantic languages (e.g., WSML, WSDL-S extending WSDL with semantic description). An example is also OWL-S used currently by the approach. An interesting aspect of this approach is that the consistency proof between WSDL description and OWL-S description of the Web service can be performed by a reasoning engine.

A similar approach to the one presented in [39], is introduced by Sun et al. in [69]. The latter aims to add semantic to the Web services described through WSDL with the objective to enable their composition. As in [39], also this solution reverse engineers Web services specified in WSDL and models them in UML. Services are then composed in UML terms. Further, the UML specification of the composed Web services is transformed in OWL. Here, a UML profile definition is part of the future work. No application of this approach to a case study is mentioned by the authors.

E. El Beggar et al.

El Beggar et al. [40] propose an approach for reverse engineering of COBOL legacy systems. The objective of this MDRE solution is to identify objects from records descriptions in three steps: extraction of data description to create the PSM models which conform to the COBOL file description metamodel, merge of the extracted PSM models and generation of a common model, and transformation of the common model into a PIM represented as a domain class diagram and refinement of the class diagram by applying heuristics that extract further information from the legacy data embedded in storage files.

The authors have validated their approach on a set of arbitrary programs written in COBOL. Furthermore, they have compared the MDRE approach with a clustering approach through three metrics: recall, precision, and F-measure. The results show that the MDRE approach performs better than the clustering one.

1) MDRE Models: A first PSM model is obtained from COBOL programs from which is extracted the data description available in the section File Description. The PSM model conforms to the COBOL file description metamodel. It is generated a PSM model for each COBOL file. The PSM models are further merged in a common model called Merge Model of File Descriptors (MMFD), which regroups the file descriptors programs. This common model represents the source model for the transformation to the target PIM model. The PIM model is represented by a domain class diagram in UML. The metamodel of the PIM conforms to the MOF (Meta-Object Facility) specification. The domain class diagram is refined with a set of heuristics which extracts information such as associations, multiplicities, objects identifiers from source files.

2) MDRE Steps: El Beggar et al. implement their approach in three steps:

- extraction of data from COBOL programs;
- merge of the data extracted from various COBOL programs into a single model;
- transformation of the single model in a domain class diagram and refinement of the last with further information extracted from storage files.

3) Tool Support: The authors mention that they exploited ATL based tools for model transformation, without providing any further details on the tools used.

4) Further Considerations: El Beggar et al. propose an automatic approach for reverse engineering COBOL programs. A particularity of this approach consists in the fact that it extracts significant data embedded in flat files to refine the obtained class model (see step 3 of this MDRE approach).

From the reusability point of view, the approach can be used only for COBOL legacy systems.

F. Favre et al.

The approach proposed by Liliana Favre et al. [41], [42], [43], [44], [51] is placed within the context of the reverse engineering of object-oriented software, according to the MDA standard. The authors focus their attention on the extraction of UML diagrams from Java source code: class and state diagrams [41], [42], use case diagrams [55], activity diagrams [87], and sequence diagrams [49]. Besides the models reconstruction, this approach proposes a formal proof of the transformations between models. With the formal proof, it is possible to maintain the consistency in the reverse engineering process.

Essentially, Favre's et al. approach exploits static and dynamic analysis to generate PSMs and PIMs from code and to analyze the consistence of the performed transformations from code to models and between models. Furthermore, the authors propose a framework in the context of which the previously mentioned transformations are performed. The framework is able to address three different abstraction levels concerning models, metamodels, and formal specifications. The authors describe their approach by exploiting simple case studies mentioned in other papers such as [88].

1) MDRE Models: In this approach, the PSM models are expressed in terms of UML (e.g., class and state diagrams) and OCL (Object Constraint Language). Metamodels describe families of implementation specific models, PSMs, and PIMs expressed in terms of MOF-metamodels. Every model conforms to a MOF-metamodel. Furthermore, the approach proposes the specification of MOF-metamodels and metamodel transformations in the NEREUS metamodeling language for consistency issues. Two types of consistency are implemented: horizontal, between models at the same abstraction level, and vertical, between different levels of refinements.

2) MDRE Steps: This approach is characterized by the following steps:

- extraction of an AST (Abstract Syntax Tree) from source code through static analysis;

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2733518, IEEE Access

- extraction of a PSM by enriching the AST with dynamically extracted information;
- abstraction of the PSM into a PIM, expressed in terms of UML diagrams.

3) Tool Support: The authors mention limited details on the techniques used for the model transformations such as classical compiler construction techniques [41], [42], [51]. Furthermore, the authors mention the use of ATL based tools for model transformation. In [49] the authors mention the use of MoDisco as a tool support.

4) Further Considerations: There are at least three advantages of this semi-automatic approach. First, it uses well known formalisms to specify models and metamodels, such as UML diagrams, OCL, and MOF. Second, it exploits both static and dynamic analysis, hence it may capture different aspects of the analyzed software. Last, it supports the consistency of the reverse engineering processes. It focuses on the formal proof of models produced through reverse engineering. The authors mention that the approach is presented for the Java programming language, but it can be reused for any object-oriented language. Being generic, it also may be applied in any application domain.

G. Fleurey et al.

Fleurey et al. [45] propose a semi-automatic round-trip model-driven engineering approach for the migration of large industrial software. The motivation behind this approach raises from the need of a full re-development of the legacy software whenever a migration of software is done. The authors sustain that a model-driven approach may increase at least partially the automation of software migration and reuse of existing design from the legacy code. This migration solution includes the automatic analysis of the source code, the generation of abstract models into target platform models, and generation of code for the target system. The approach has been validated on an actual case study, written in COBOL, which implied the migration of a large scale banking system from mainframe to J2EE.

1) MDRE Models: This approach builds the AST from the source code. Further, from the AST the code model is obtained. The code model representing the PSM conforms to the metamodel of the legacy programming language. A PIM is obtained through model-to-model transformations. The PIM model conforms to the ANT metamodel [45]. ANT may represent static data structures, actions and algorithms, graphical user interfaces and widgets, and application navigation.

2) MDRE Steps: The migration process proposed by this approach consists in the following steps:

- extraction of the meaningful information from source code to build a PSM model;
- transformation of the code PSM model into a PIM conforming to the ANT metamodel;
- transformation of the ANT model into a PSM for the target application;
- generation of the code for the new application from the PSM model.

3) Tool Support: The authors provide a tool suite called Model-In-Action (MIA) for this approach. The tool suite is implemented for the round-trip engineering being composed of a MIA-Transformation module for model-to-model transformations, and a MIA-Generation module for code generation.

4) Further Considerations: Fleurey et al. propose a generic round-trip engineering model-based approach. For the case study, the authors have validated the migrated application through a strict non-regression testing process, which has represented 45% of the total project costs. The reason behind this validation is the semi-automatic aspect (i.e., the manual executed tasks may introduce errors).

H. Garcia-Rodriguez de Guzman et al.

Garcia-Rodriguez de Guzman et al. [47], [89] propose an ADM based approach, called PRECISO, for recovering Web services from legacy databases because the last represent a valuable asset for any organization. The objective is to expose data stored in legacy databases through Web services. For this approach, the authors have exploited their experience in database re-engineering [34], [59], [90].

The authors have validated their approach through a case study concerning a joint project between the University of Castilla-La Mancha and Indra Software Labs [89]. The case study is a corporate portal for the management of the information produced from the collaboration between industry and university (e.g., conferences, lectures, courses, grants, events, awards, publications).

1) MDRE Models: The first model extracted from the source artifacts is the database model. The authors consider relational databases, hence the obtained database model conforms to the SQL-92 metamodel. Based on the relations between the database tables expressed through primary and foreign keys, the authors extract a first list of candidate services.

The second model generates an object model according to UML metamodel from the relational database schema.

Finally, the WSDL-based model of the Web services is obtained from the object model. The WSDL model is part of the forward engineering phase, hence out of the scope of this paper.

2) MDRE Steps: Essentially, the approach can be summarized in three main steps:

- the extraction of the database model from the legacy system, which generates a platform dependent model of the database and identifies an initial list of potential candidates as services.
- the generation of a UML object model from the database model;
- the generation of Web services to manage the legacy database, part of the forward engineering phase.

The first step has two sub-steps: the extraction of the PSM model and the discovery of the potential service candidates. The PSM model concerning the relational database schema is obtained through a set of homogeneous queries on the information schema of the legacy database. Further, the PSM is made persistent by using XMI. The list of the service candidates is obtained through model-driven pattern matching which capture the relationships between business entities and are expressed through foreign keys.

3) Tool Support: The authors provide the PRECISO tool support as a stand-alone semi-automatic desktop application. The tool supports the creation of Web Services from legacy relational databases.

4) Further Considerations: The advantage of this approach results from its objective: the extraction of services from the legacy databases. It is based on the database persistent information rather than on the business application.

I. Lenk et al.

Lenk et al. [48], [91] propose a round-trip engineering approach for the development of 3D Web applications based on X3D and JavaScript. In the forward phase, an abstract model expressed in the SSIML (Scene Structure and Integration Modeling Language), a domain-specific language, may be automatically transformed into X3D, JavaScript, or C++ code. In the reverse engineering phase, the manual modifications performed by the developers to the previously generated code are identified and merged into the abstract SSIML-based model.

The approach has been applied to various applications on different platforms such as Web, immersive virtual reality, and augmented reality on mobile devices.

1) MDRE Models: This approach uses the following models: a SSIML abstract model, intermediate models, and ASTs. SSIML is a domain specific language for the definition of 3D applications. An intermediate model represents a central data structure in this approach. It describes the SSIML, JavaScript, and X3D contents of a Web application in a generic way. An intermediate model is not intended to be viewed or edited by developers. The approach uses two ASTs: one for the JavaScript code and one for the X3D code.

2) MDRE Steps: The reverse engineering phase of this round-trip approach consists in the following steps:

- extraction of the JavaScript and X3D ASTs from source code using Xtext;
- generation of the intermediate model from the ASTs through ETL (Epsilon Transformation Language), a hybrid transformation language with declarative rules and imperative constructors;
- merge of the modifications manually performed in the source code into the initial intermediate model (generated in the forward phase);
- extraction of the SSIML abstract model through Java with reflective API provided by the Eclipse Modeling Framework.

3) Tool Support: The authors exploit a SSIML-based tool which supports the creation of SSIML models and integrates round-trip capabilities.

4) Further Considerations: One of the advantages of this approach concerns the fact that it considers source code written in different languages: JavaScript and X3D. Further, by introducing the intermediate models, the approach can be extended to other programming languages (i.e., authors mention C++). From the reusability point of view, the mapping between the intermediate model and SSIML can be reused, being independent of any programming language.

J. Normantas and Vasilecas

Normantas and Vasilecas [52] propose an approach for business rules and business scenarios extraction from existing software systems with the objective to automate the systems comprehension and reduce maintenance costs. In [53], the authors address issues concerning the extraction of knowledge from software artifacts and the representation of the extracted knowledge into the KDM metamodel with the objective to abstract the business logic implemented in a system.

The approach has been applied on a commercial-off-the-shelf enterprise content management system used in several governmental organizations.

1) MDRE Models: Normantas and Vasilecas exploit various KDM models at different abstraction levels to represent information extracted from enterprise systems.

The Inventory model, at the Infrastructure layer, captures physical artifacts (source files, folders, containers, resource definitions) and represents them at a higher abstraction level. This model is created in an early stage through file system scanning or version control system querying.

The Code model, at the Program Elements layer, captures structural and behavioral information about the analyzed system. The structural information is extracted directly from AST, while behavioral information need numerous AST traversals to discover the data and flow controls.

At the Runtime Resource layer, four models are used: UI, Data, Platform and Events. The approach captures runtime objects and establishes the dependencies between them and source code objects through the Code models.

The authors create also a database of software documentation from digital documents (e.g., libraries, word documents).

As Cosentino et al. [84], this approach enables the traceability of the implementation of the business rules and business scenarios in the software under analysis.

2) MDRE Steps: This approach is implemented through three main steps:

- a preliminary study, which gathers the initial information about the enterprise system and defines a strategy to extract the necessary knowledge from the system;
- knowledge extraction, which discovers the knowledge and creates the KDM-based models;
- business logic abstraction, which aims to separate KDM model parts representing business logic implementation from the infrastructure parts.

3) Tool Support: As tool support for their approach, the authors exploit the Eclipse platform and the available tools implementing the KDM framework in Eclipse.

4) Further Considerations: This approach is a semi-automatic one based on static analysis. It is a generic approach that can be applied to various enterprise software systems. As future work, the authors plan to express the KDM models also as UML models to facilitate the understanding and application of their solution.

K. Perez-Castillo et al.

Perez-Castillo et al. [92] proposes a semi-automatic approach to extract the business processes from legacy information systems. The approach is based on the MARBLE (Modernization Approach for Recovering Business processes from LEgacy Systems) [56], [93] generic and general-purpose framework which discovers the business process from legacy systems using ADM and KDM. This solution includes the extraction of knowledge from legacy systems through a static and dynamic analysis, the generation of the KDM model from the extracted knowledge through QVT, and the discovery of the business processes from the KDM model through the application of business patterns and QVT model transformations.

The authors have applied their solution on various industrial case studies in various application domains [56]: healthcare, e-government, and enterprise systems. In [92], the authors present in detail the VillasanteLaboratory case study, a legacy system which manages the operation of a Spanish company acting in the water and waste industry.

1) MDRE Models: The first models generated from the legacy system are obtained through a static and dynamic analysis of the source artifacts. The static analysis leads to the building of the ASTs. A PSM model is generated for each of the software artifact (i.e., code, interface). This approach analyzes Java-based systems, and therefore the PSMs conform to the Java metamodel. The dynamic analysis of the legacy systems enriches the PSMs with log files obtained during the execution of the legacy system.

The approach generates a PIM as an integrated view of all the PSM models. The PIM is a KDM-based model. This model works as a KDM repository. It uses the packages Code and Action from the KDM Program Element Layer.

The higher level abstraction model is a business process model extracted from the KDM based representation. It is obtained by following business patterns and experts indications. It conforms to the BPMN metamodel, which represents flow object, connecting object, artifact, and swim lane elements.

2) MDRE Steps: The reverse engineering process proposed in this approach is composed of the following main steps:

- extraction of the PSMs from each legacy software artifact;
- transformation of the PSMs into a KDM-based PIM implemented by means of QVT relations;
- transformation of the KDM model into a business process model.

3) Tool Support: The authors use MARBLE, an extensible framework based on ADM. It is implemented as an Eclipse plug-in. It enables the interaction with experts during the extraction of business processes, hence being a partially automated model-driven reverse engineering approach.

4) Further Considerations: Perez-Castillo et al. sustain that their approach, known as the Business Process Archeology method, has the following three advantages:

- being an ADM-based approach, its reusability, formalization, repeatability, and automation are enabled;
- exploiting KDM, the business knowledge management is performed in an integrated and standardized manner;
- maintenance and traceability issues are enabled, because MARBLE identifies what elements of the business process are obtained from specific pieces of legacy code.

L. Ristic et al.

Ristic et al. [62] propose a model-driven database reengineering approach, which comes to complement the code reengineering in data-oriented software systems. Generally, a relational database reengineering has two main phases: data structure extraction and data structure conceptualization. The authors sustain that the data structure extraction is hardly achievable through a fully model-driven approach because of often user interaction needs, hence their model-driven solution is focused mainly on the data structure conceptualization.

The approach has been validated through a toy example considering a university database [62], [63]. The database is implemented under the Oracle management system.

1) MDRE Models: The PSM model generated during the data extraction phase is expressed through a vendor-specific physical database schema. It captures information about data types, tables, columns, check constraints, primary key, and unique key constraints accompagned with foreign key constraints. This model is semantically enriched with further information extracted (i.e., inverse referential constraints and homonym inconsistencies) from the legacy database and through interaction with designers. The PSM model is an XML document.

The PIM model represents the desired conceptual database model and belongs to the data structure conceptualization phase. The approach allows the creation of different PIMs which may conform to the following metamodels: Enhanced Entity-Relationship (EER), Class, or Form Types (FT). The discussion in [62] concerns a PIM conforming to FT. Also PIM is an XML document.

2) MDRE Steps: The reverse engineering process proposed in this approach is composed of the following main steps:

- metadata extraction from a relational database and its storage in IIS*REE repository (i.e., a tool for reverse engineering, part of the IIS*Case tool [62]);
- semantic enrichment of the extracted metadata, implemented in OracleJ Developer environment;
- XML generation of the PSM model;
- XML2RDBMS, transformation of the PSM model conforming to XML metamodel into a model conformant with an SQL standard metamodel;
- RDBMS2RM, transformation of the model conformant with a SQL standard metamodel into a model conformant with the generic relational database metamodel;
- RM2IISCase, transformation of the model conformant with the generic relational database metamodel into a model conformant with the FT metamodel;
- IISCase2XML, transformation of the model conformant with the FT metamodel in an XML document.

3) Tool Support: This approach uses as tool support the IIS*Studio development environment. IIS*Studio has a IIS*Case tool for forward engineering, and a IIS*REE tool for reverse engineering, which has been exploited for the implementation of this MDRE approach.

4) Further Considerations: This approach is part of a round-trip model-driven database engineering solution. Hence, the obtained database model may be used further to generate a database schema of an information system. The authors mention that the extracted FT specifications can be enriched with the specifications of transactions and business applications and further used to generate code and appropriate GUI for the management of information available into a database.

M. Rodriguez-Echeverria et al.

The MIGRARIA project [64], [94] aims to define a semi-automatic process to modernize legacy non-model-based data-driven Web applications into Rich Internet Applications (RIA) following the OMG ADM guidelines. RIA emerged as a promising platform for Web 2.0 development. The reverse engineering approach extracts conceptual models from legacy Web applications developed using Web development frameworks based on the Model-View-Controller (MVC) pattern. The main steps of this approach are: extraction of technology-dependent models from source code through static analysis, generation of the conceptual MVC-based models, and transformation of MVC models into Model-Driven Web Engineering (MDWE) models. Generally, MDWE approaches provide models for presentation, navigation, and data concerns from a conceptual point of view.

In [95], [96], the authors describe how navigational models are extracted from legacy Web applications through their approach. In [97], the authors explain how to generate a REST API from legacy Web applications, starting from the conceptual model of the legacy application obtained through their reverse engineering approach. In [68], the authors indicate how their approach can be used to evolve legacy Web applications based on the MVC pattern towards a service oriented architecture.

This approach has been validated through the Agenda system, an example of a data-driven Web application for the management of the students' agenda in a university [96], and through a Conference Review System [97].

1) MDRE Models: Rodriguez-Echeverria et al. define the MIGRARIA MVC metamodel, which specifies the main concepts of the development of Web applications based on the three main components of the MVC pattern. The Model component represents data objects, their attributes, their relationships, and the operations defined over them. The View component represents pages as main containers, and presentation objects and requests as main contents. The Controller component represents request handlers, the mappings between presentation and data objects, their response defining a relationship with the target page element, and the sequence of operation calls performed to execute an action or to fetch data. Finally, the MVC based model is transformed in a target MDWE model to obtain a conceptual representation of the legacy Web application in a concrete MDWE approach. In [98], the authors mention among the technology-dependent models those specific for the JSP, XML, and Java languages. The MVC conforms to Struts metamodel. The target MDWE language is WebML.

2) MDRE Steps: The approach proposed by Rodriguez-Echeverria et al. implements three main steps:

- extraction of technology-dependent models from source code;
- generation of a conceptual MVC model;
- transformation of the MVC model into a MDWE model.

3) Tool Support: The authors use MoDisco discoverers to generate the PSM model from source code. They also extended MoDisco for Struts MVC models [95]. The PIM model is implemented through ATL based support [95], [96].

4) Further Considerations: The authors outline that they use common model-driven methods, techniques, and tools to make their approach a systematic, replicable, and reusable process.

N. Sanchez Ramon et al.

Sanchez Ramon et al. [65], [66], [67] describe a model-driven reverse engineering approach of Graphical User Interfaces (GUIs) of systems developed with Rapid Application Development (RAD) tools. This approach is intended to modify (e.g., migrate applications to advanced platforms) or adapt (e.g., to mobile devices) GUIs. In the RAD systems, a GUI layout is implicit, thus each object, which is part of the interface, has associated spatial coordinates. To enable the evolution of these GUIs, high level representations of them can be generated and a model-driven based approach can be applied. Models are used at a high abstraction level to make explicit the GUIs layout.

The approach has been validated through two real applications in two different application domains created by different developers. The objectives of these two case studies were (1) to apply the reverse engineering approach to the available GUIs and (2) to generate Java Swing user interfaces.

1) MDRE Models: This approach for legacy GUIs aims to express GUIs in terms of Concrete User Interface models conforming to the specification introduced in [99]. The layout of the GUIs elements in such models is expressed through relative positions of the elements among them, and not through absolute coordinates.

To obtain a Concrete User Interface (CUI) model from a legacy GUI, four intermediary models are created. First, from the legacy GUI a Source Technology model is extracted. This model depends of the RAD specific technology used to develop the GUI. Further, a normalization of the Source Technology model leads to the definition of a RAD model, which is technology independent. The RAD model is annotated with additional layout information in two stages. After a first stage a Region model, which makes explicit the visual containment relationships among widgets, is obtained. During the second stage a Tile model, which refines regions and eliminates elements positions expressed through coordinates, is generated. The Tile model expresses the layout of the GUIs elements through relative positions among them rather than using coordinates. The CUI model with explicit high-level structures, is created from a Tile model. The approach is based on two metamodels: RAD and CUI.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2733518, IEEE Access

Finally, the CUI representing the original or updated user interface may be moved to a different technology obtaining a Target Technology Model, from which the user interface code may be automatically generated. This last model is part of a forward engineering process, which is out of the scope of this paper.

2) MDRE Steps: The reverse engineering process proposed in this approach is a waterfall one composed of the following steps:

- extraction of the Source Technology Model from source code;
- transformation of the Source Technology Model into a RAD model;
- annotation of the RAD model and creation of the Region Model;
- annotation of the Region Model and creation of the Tiles Model;
- transformation of Tiles Model into a Concrete User Interface model.

3) Tool Support: The authors propose a tool called UsiResourcer [67], that reverse engineers automatically MS Windows resource files into a CUI model. For the implementation of the models transformation, the approach uses the RubyTL language.

4) Further Considerations: The main advantage of this approach is that it addresses well-defined types of applications, and thus it focuses on specific issues. It allows the reverse engineering of GUI without the need of considering the whole system. A further advantage is the fact that this solution can be adopted and adapted to further issues of RAD-based applications, such as event handling [66] or navigation flows [65]. It is hardly reusable for other types of applications. Also, it uses non standard metamodels. This approach is based on the authors' experience with re-engineering user interfaces as described in [35], [100].

O. Trias et al.

The approach proposed by Trias et al. [70], [71], [72], [73] aims to automate the migration of Web applications to CMS (Content Management System) based Web applications. CMS provides support for multiple users with different permissions levels to manage large amount of digital content. Available Web applications are migrated to CMS-based Web applications by exploiting the ADM concepts to automate the process.

The approach has been validated through a case study which migrates a wellness and nutrition Web application called Websana to a CMS-based Web application implemented in Drupal [73]. The Web application is written in PHP.

1) MDRE Models: The approach proposed by Trias el al. is starting from PHP code. The first model extracted from PHP is an AST which conforms to the ASTM_PHP metamodel, i.e., a Specific ASTM for the PHP programming language defined by Trias et al.

The authors exploit the KDM models for the PIM level. More precisely, they use the Code package to represent the items extracted from source code and structural relationships among them, and the Action package, to represent the behavior, control and data-flow relationships [73].

Finally, the approach generates the CMS models for the application migration. These models is out of the scope of this paper.

2) MDRE Steps: The reverse engineering process of this approach is composed of the following steps:

- knowledge extraction, from PHP code using Xtext and EMF;
- KDM models generation from ASTM_PHP models obtained through model-to-model transformations implemented in ATL;
- CMS model automatic generation for Drupal.

3) Tool Support: In [74], the authors present their toolkit called RE-CMS, which supports their approach.

4) Further Considerations: The solution targets the Web applications written in the PHP programming language. Thus, from the reusability point of view, the approach can be adopted for other PHP Web applications.

P. Warwas and Klusch

Warwas and Klusch [75] propose a semi-automatic MDRE approach for multi-agent systems. Their objective is to extract from source code the design of Belief, Desire, Intention (BDI) agents in order to make it reusable on a platform independent layer. The approach uses a domain specific modeling language called Domain Specific Modeling Language for Multiagent Systems (DSML4MAS) to represent agents' artifacts, which are stored in a model repository.

The approach has been validated on the open source Jadex BDI agent platform and on a real world scenario, i.e., the Mars World Classic (MWC) application example of the Jadex 2.0 RC6 distribution.

1) MDRE Models: The approach, as described in [75], is specific to the Jadex platform. Therefore, metamodels and models are thought for Jadex applications. A Jadex application uses XML to specify agents and their features and Java-based behavior. The approach uses four metamodels: the Jadex metamodel, the Java metamodel, the Jadex Project metamodel, and the PIM4Agents metamodel. The Jadex metamodel is based on the Jadex XML schema files. The Java metamodel is used to extract the Java behavior from the source code and represent it into models. The Jadex Project metamodel aggregates all the resources (e.g., agents, features, behavior) of a Jadex application. The PIM4Agents metamodel is the target metamodel for this reverse engineering approach. PIM4Agents defines the abstract syntax for the DSML4MAS.

The approach uses two main models: the project model, which conforms to the Project metamodel, and the information model, which conforms to the Jadex and Java, respectively, metamodels. The information model(s) capture the description of the agents, their features, and their behavior.

The main objectives of this approach is to represent multi-agent systems developed in Jadex through the DSML4MAS. The Jadex-based metamodels cover similar concepts to the PIM4Agents metamodel. The authors define ten model transformation rules from Jadex and Java concepts to PIM4Agents concepts [75].

2) MDRE Steps: The reverse engineering process proposed in this approach is composed of the following steps:

- create automatically a Jadex Project model and information model(s), which contain all the resources of a Jadex example project;
- apply ten transformation rules through the Jadex4PIM4Agents on the models obtained in the previous step using QVT;
- manual refinement of the models to specify the communication among agents.

3) Tool Support: The generation of the PSM model is implemented using EMF (Eclipse Modeling Framework) and MoDisco. The model transformations are implemented through QVT based support.

4) Further Considerations: Warwas and Klusch propose a reverse engineering approach for multi-agent systems. It can be reused in multi-agents applications. One of the advantages of this approach concerns the model validation. The semantic of the DSML4MAS has been formally defined in Object-Z and transferred to the OCL-based constraints for validating PIM4Agents models. The model validation at a platform independent level may prevent many errors in early stages of a project. Furthermore, the approach uses a model repository containing validated artifacts which can be reused during the reverse engineering of various multi-agent systems.

IV. Comparison of the MDRE Approaches

The main features of the considered approaches, i.e., the models exploited, the scope of the approach, the used tools, the case studies analyzed, the automation level of the analysis process, and the type of legacy source analysis, that we introduced, at the beginning of the previous section to provide a comparative framework, are summarized in Table III.

In the *Models* column, two types of information co-exist: the models' names used by an approach (e.g., Cosentino et al.) and the paradigm used to define the models (in italic), when the approaches do not indicate a name for each model they create and specify the paradigm used to define the models (e.g., Bruneliere et al.).

The *Scope* column indicates if the approach is generic or specific. A generic approach indicates methods or infrastructure to support the definition of new MDRE processes, or defines a process supporting a precise goal, but independent from single technologies or applications domains. A specific approach, conversely, is defined to apply MDRE to solve a problem related to a specific technology, domain, or case study. A MDRE specific approach has often the chance of being generalized, through the use of general metamodels (and associated transformations), and become a generic approach.

The *Tools* column mentions the tools used to implement the reverse engineering approach.

The *Case Study* column outlines the available case studies where the approach has been applied.

The *Auto* column indicates the automation level of the MDRE approaches, i.e., automated (T) or semi-automated (P).

The *Type* column indicates the type of the analysis performed by each MDRE approach on the source artifacts, i.e., static (S), dynamic (D), or both.

The answers to RQ1, RQ2 and RQ3 in the next section cover all the features related to Models, Tools and Automation level.

From the Type of analysis point of view, we observe that twelve out of fifteen MDRE approaches are based only on a static analysis of the legacy system (Cosentino et al., Djamel et al., El Beggar et al., Fleurey et al., Garcia-Rodriguez de Guzman et al., Lenk et al., Normantas and Vilecas, Ristic et al., Rodriguez-Echeverria et al., Sanchez Ramon et al., Trias et al., Warwas and Klusch). Only three approaches use also dynamic analysis to complement the static analysis of the legacy artifacts (Bruneliere et al., Favre et al., Perez-Castillo et al.). The first exploits execution traces to generate dynamic views of a legacy system (e.g., UML sequence diagrams). The second generates dynamic views of a legacy system by exploiting debuggers, event recorders, and general tracer tools. The third enriches the PSM statically obtained with further information collected in log files during the execution of the legacy system. No approach is based only on dynamic analysis of legacy systems.

The Scope aspect is discussed below according to the Genericity feature proposed by Bruneliere et al.

In the remaining of this section, we briefly discuss the MDRE approaches according to the four features of Bruneliere et al. introduced at the beginning of Section 3.

a) Genericity: Through our SLR, five out of fifteen approaches have a generic scope. Actually, only one of these five approaches, i.e., Bruneliere et al., provides various case studies in different application domains with various objectives. Favre et al. does not provide any case study. Fleurey et al. provides one case study in the banking domain, Normantas and Vasilecas various examples, all of them concerning commercial-off-the-shelf enterprise content management examples, while Perez-Castillo et al. mention three case studies in different domains with the objective to extract the business process they implement.

The remaining ten approaches have a specific scope, being designed for well-defined objectives or for specific types of systems. Four out of these ten specific approaches concern various aspects of Web-based legacy software, while five of them concern Java, COBOL, RAD, relational databases, and multi-agents systems.

b) Extensibility: Some MDRE approaches (Djamel et al., El Beggar et al., Rodriguez-Echeverria et al., Sanchez-Ramon et al., Trias et al., and Warwas and Klusch) describe the link between the reverse engineering steps and the further forward engineering (based on the models obtained through reverse engineering). Essentially, these MDRE approaches have the objective to modernize existing legacy systems, and hence to obtained a model-driven version of the existing legacy system. Further, all these MDRE approaches have a specific scope. Fleurey et al., Garcia-Rodriguez de Guzman et al., Lenk et al., and Ristic et al. combine reverse and forward engineering, and present their approaches as round-trip model-driven solutions. Cosentino et al. link their approach with forward engineering by providing textual and graph presentations of the results to facilitate the comprehension and further exploitation of the extracted business rules.

TABLE III: MDRE Approaches

MDRE Approach	Models	Scope	Tools	Case Study	Auto	Type
Bruneliere et al. 2014	• *KDM* • *SMM* • *ASTM* • *Java* • *JSP* • *XML*	G	• MoDisco	• Amadeus Hospitality (migration from VB6 to JEE) • Sodifrance (modernization of projects) • ARTIST	T	S,D
Cosentino et al., 2012	• Java • Domain Variable Model • Business Rule Model • Global Domain Model • Generic/Specific Graph Model	S	• MoDisco • ATL	• A simulator of the living beings behavior • IBM case study	T	S
Djamel et al. 2008, Sun et al 2009	• *UML profile (UML class and activity diagrams)* • *OWL-S service description*	S	• Protégé, • UML tools, • Conversion tools to map UML→OWL-S	CongoBuy	P	S
El Beggar et al. 2013	• COBOL • MMFD • UML Domain Class Model	S	• ATL	Arbitrary COBOL programs	T	S
L. Favre et al. 2008, 2009, 2011, 2014	• *NEREUS* • *UML* • *OCL*	G	• ATL • MoDisco	N/A	P	S,D
Fleurey et al. 2007	• *AST* • *ANT* • *UML*	G	• MIA	Large Banking System: from Mainframe to J2EE	T	S
Garcia-Rodriguez de Guzman et al., 2007	• SQL-92 Model • Relational Database Model • Object Model	S	• PRECISO	Intranet Java-based system	P	S
Lenk et al. 2015	• SSIML • IM • X3D AST • Javascript AST	S	• SSIML Ad Hoc tool	3D mobile-desktop applications	P	S
Normantas and Vasilecas, 2012	• KDM Models: Inventory, Code, UI, Data, Platform, Events.	G	• ATL • Eclipse tools for KDM	Commercial-off-the-Shelf Enterprise Content Management	P	S
Perez-Castillo et al., 2011	• KDM Code/Action • Java • BPMN	G	• MARBLE • QVT	• Healthcare system • E-Government system • Enterprise system	P	S,D
Ristic et al. 2015	• XML • SQL • generic relational DB	S	• ATL	• University DB	P	S

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2733518, IEEE Access

MDRE Approach	Models	Scope	Tools	Case Study	Auto	Type
Rodriguez-Echeverria et al. 2013	• MVC MM • Java • JSP • XML	S	• MoDisco, • ATL	• Agenda, • Conference Review System	P	S
Sanchez-Ramon et al. 2010, 2011, 2014	• Source technology model • Rapid application devel. model • Region model • Tile model • Concrete user interface model	S	• RubyTL • UsiResourcer	Two Oracle Forms for Spanish applications (for managing research projects and grants, and for managing regional financial and organizational aspects)	T	S
Trias et al. 2015	• ASTM_PHP • KDM Code-Action	S	• RE-CMS	Websana	T	S
Warwas and Klusch 2011	• Java • Jadex • PIM4Agents	S	• MoDisco • QVT	Jadex Mars World Classic	P	S

Scope:G=Generic,S=Specific. Auto(mation Level of the RE Process):T=Total,P=Partial. Type(of Source Code Analysis):S=Static,D=Dynamic.

Finally, all the generic approaches (Bruneliere et al., Favre et al., Normantas and Vasilecas, and Perez-Castillo et al.) have as main objectives the representation of the legacy systems at a higher abstraction levels through models conforming to metamodels. They do not mention any further usage of the created models. However, the obtained models may be further manipulated and used in MDE steps.

All the MDRE approaches considered in this survey ensure the decoupling between the reverse and the forward parts of a round-trip model-driven solution independently on the fact that they are generic or specific approaches.

c) Automation: We tried to established a correspondence between the automation level and the scope of a MDRE approach. Actually, we found no rules because there are generic approaches totally automated, i.e., Bruneliere et al. and Fleurey et al., and generic approaches partially automated, i.e., Favre et al., Normantas and Vasilecas, and Perez-Castillo et al. The same observation holds also for the specific approaches: four are totally automated, while six partially automated.

d) Full or partial coverage: Generic MDRE approaches (i.e., Bruneliere et al., Favre et al., Fleurey et al., Normantas and Vasilecas, and Perez-Castillo et al.) tend to have a full coverage of the source artifacts. The specific approaches are divided between full and partial coverage of the source artifacts. For example, Djamel et al., El Beggar et al., Lenk et al., Rodriguez-Echeverria et al., Trias et al., and Warwas and Klusch have a full coverage of the source artifacts.

Four of the specific MDRE approaches are focused on the parts relevant for their objectives. For example, Cosentino et al. extract business rules by isolating the code segments concerning the business processes. Garcia-Rodriguez de Guzman et al. and Ristic et al. focus on the information available in legacy databases. Sanchez-Ramon et al. outline that their approach allows the reverse engineering of GUI without the need of considering the entire system; in this last case, the partial coverage of the system is considered as an important advantage, and not a limitation. We observed that these four specific approaches are totally automated.

e) Direct (re)use and integration: Generally, for the specific MDRE approaches, the range of the reusable and/or integrated elements is more limited than in the generic approaches and those based on standards; on the other hand, they are focused and optimized for a well-defined goal.

For example, Cosentino et al. approach enables the reuse of the domain model and of the extracted business rules (i.e., as auxiliary information) for other applications in the same domain. Djamel et al. approach can be reused for other semantic languages. El Beggar et al. approach is reusable for COBOL legacy software. Garcia-Rodriguez de Guzman et al. approach can be reused for the modernization of relational databases towards a Web services solution. Lenk et al. round trip engineering approach is suitable for 3D Web applications. Ristic et al. approach can be reused for relational databases. Rodriguez-Echeverria et al. approach is suitable for Web applications developed based on the MVC pattern. Sanchez-Ramon et al. approach can be reused for other issues of RAD-based applications. The approach proposed by Trias et al. is reusable for Web applications having also the advantage of being based on KDM models. Warwas and Klusch approach may be exploited for reverse engineering multi-agent systems.

The generic approaches reuse and integrate available elements starting from standards, metamodels, models (e.g., KDM, UML profiles), or development environments (e.g., Eclipse). At the same time, their processes and results can be further used by various systems in different domains and implemented in different programming languages. For example, Bruneliere et al. offer a common MDRE approach for many reverse engineering scenarios. Favre et al. define an approach for object-oriented legacy systems. Fleurey et al. are focused on large systems in the banking application domain. Normantas and Vasilecas aim to address reverse engineering issues of enterprise systems. Perez-Castillo et al. extract business processes from any legacy system.

V. Results and Analysis

In this section we provide the answers emerged from our survey to the research questions introduced in Section II.

A. RQ1

Which metamodels are used by the model-driven reverse engineering approaches? Are they defined to solve specific problems or are they reused for more than one purpose?

1) Results: From the description of the approaches presented in Section III and summarized in Table III, we can observe a significant heterogeneity in the metamodels used by the MDRE solutions. A possible categorization may group the MDRE approaches as:

- standardized models, i.e., KDM, and modeling languages, i.e., UML;
- ad-hoc models (Cosentino et al., Djamel et al., El Beggar et al., Favre et al., Fleurey et al., Rodriguez-Echeverria et al., Sanchez Ramon et al.);
- available domain-specific models (Lenk et al., Warwas and Klusch).

There are four MDRE approaches out of fifteen which reuse and extend the KDM metamodels (Bruneliere et al., Normantas and Vasilecas, Perez-Castillo et al., and Trias et al.). For example, Normantas and Vasilecas exploit KDM for different purposes, i.e., to model the code, the user interfaces, and business rules, and thus, KDM extensibility features have been applied.

Three out of fifteen MDRE approaches exploit UML using its default profile (Favre et al.) or adding customized profiles (Djamel et al.), and define UML based models specific to a particular task or domain (El Beggar et al., Favre et al.). All these three approaches define new metamodels.

Two of the described MDRE approaches deal with relational databases and generate a SQL model and a generic relational database model (i.e., Garcia-Rodriguez de Guzman et al. and Ristic et al.).

There are four MDRE approaches which define new ad-hoc metamodels (Cosentino et al., Fleurey et al., Rodriguez-Echeverria et al., and Sanchez Ramon et al.). Currently, the reuse of these metamodels are limited to other works of the same (or similar group) of authors. For example, the approach proposed by Sanchez Ramon et al. for GUIs developed with RAD tools has been reused for reverse engineering of event handling or navigation flows in RAD applications (see Section III).

Two of the fifteen approaches reuse domain-specific models: Lenk et al. in context of 3D Web applications, while Warwas and Klusch in the context of multi-agent systems.

To summarize, eight MDRE approaches reuse existing metamodels, while seven define new ones.

2) Analysis: Four MDRE approaches use KDM metamodels. It will be interesting to investigate if other approaches and models can be integrated with KDM in order to reach a common and uniform representation of the analyzed systems. For example, Normantas and Vasilecas exploit KDM for various purposes among which to model the user interfaces and business rules. Sanchez Ramon et al. define new models to represent user interfaces and Cosentino et al. to represent business rules. A future issue may investigate the feasibility of integrating these last two specific models with KDM, asking the authors of these MDRE approaches if KDM has limits that do not allow a real integration with models addressing specific purposes, or if there are significant advantages of defining new models for specific purposes.

Given the apparently limited reuse of standardized models (e.g., KDM) for specific case studies, we have identified a possible cause in the specification of metamodels. Standard and organic models as KDM, SMM, and ASTM have very large specification documents, filling hundreds of pages. This factor may discourage their application, while it is simpler in general to incrementally define an ad-hoc metamodel meeting specific requirements. Moreover, the main implementation of KDM, which comes with the MoDisco project, had traditionally problems in handling large models, as reported by MoDisco authors [36], limiting the application of these tools to enterprise-class software. The underlying EMF framework, in fact, gained the possibility of handling very large models transparently through a dedicated storage solution (CDO[3]), mapping to a relational database (Teneo[4]) or to a graph database [101] some years after its creation, and only recently the implementation and the average computational power available have become sufficient to handle very large models.

B. RQ2

Which tools are used by the approaches for their implementation? Do the approaches provide new tools or do they re-use existing tools?

1) Results: From the description of the approaches presented in Section III and summarized in Table III, it results that all the MDRE approaches have associated tool support. A possible categorization of the MDRE approaches may group MDRE approaches in those using:

- new tools created in the context of an MDRE approach;
- existing tools defined in the context of other MDRE approaches or for other purposes.

[3]http://www.eclipse.org/cdo/

[4]http://wiki.eclipse.org/Teneo#teneo

Seven out of fifteen MDRE approaches define new tools for implementing their approach (Bruneliere et al., Fleurey et al., Garcia-Rodriguez de Guzman et al., Lenk et al., Perez-Castillo et al., Sanchez-Ramon et al., Trias et al.). One of these new tools is MoDisco developed by Bruneliere et al.

Eight out of fifteen MDRE approaches use available tools for implementing their approach (Cosentino et al., Djamel et al., El Beggar et al., Favre et al., Normantas and Vasilecas, Ristic et al., Rodriguez-Echeverria et al., Warwas and Klusch). Four of these nine MDRE approaches (Cosentino et al., Favre et al., Rodriguez-Echeverria et al., Warwas and Klusch) use MoDisco for their implementation, while six of them use ATL-based tools.

2) Analysis: Even if all the MDRE approaches have associated tool support, we observed that most of the authors just mention briefly which tools are used in their solution without providing details on their implementation and exploitation. In the recent years however, we noticed that the authors belonging to research groups which have a continuous interest in MDRE have developed their tools and published more details on tool support: MoDisco by Bruneliere et al. in 2010, MARBLE by Perez-Castillo et al. in 2011, and RE-CMS by Trias et al. 2015,

In our opinion, the tool support plays a main role in the spread and reuse of a MDRE approach. For example, it seems that the spread of KDM was eased by MoDisco, as the latter provides a Java/EMF implementation of the models, and provides discoverers for extracting models from source code. We expect a larger adoption of KDM and the other ADM models in the future with the increase of tool support.

C. RQ3

What is the level of automation of the transformations defined in the MDRE approaches?

1) Results: From the description of the approaches presented in Section III and summarized in Table III, it results that the reverse engineering process may be:

- partially automated, meaning that human intervention (e.g., software engineering experts) is needed during one or more steps of the approach execution;
- totally automated, meaning that no human intervention is needed during the approach execution.

Nine out of fifteen MDRE approaches are partially automated (Djamel et al., Favre et al., Garcia-Rodriguez de Guzman et al., Lenk et al., Normantas and Vasilecas et al., Perez-Castillo et al., Ristic et al., Rodriguez-Echeverria et al., Warwas and Klusch). The human intervention may be needed to refine or enrich the obtained models with further information (Djamel et al., Garcia-Rodriguez de Guzman et al., Normantas and Vasilecas, Ristic et al., Rodriguez-Echeverria et al., and Warwas and Klusch), to interact during a dynamic analysis of source artifacts (Favre et al., Perez-Castillo et al.), or to solve differences between extracted and generated models in round-trip approaches (Lenk et al.).

Six out of fifteen MDRE approaches are totally automated (Bruneliere et al., Cosentino et al., El Beggar et al., Fleurey et al., Sanchez Ramon et al., and Trias et al.). Four of these approaches are designed for specific objectives and for specific application domains. Fleurey et al. sustain that their approach has a general purpose (being classified as generic in Table III); however, its only application up to now has been in the banking domain. Bruneliere et al. propose a general purpose approach and the authors have provided several case studies in various application domains.

2) Analysis: One of the objectives of the MDRE approaches is to automate as much as possible the reverse engineering process. There are at least a couple of reasons behind this objective. First, the automation avoids human intervention, which may introduce errors in the generated models. Second, the automation of the MDRE process may lead to its further adoption in other case studies and application domains. The results of our survey show that almost half of the considered MDRE approaches are totally automated. This automation result may be due to the growing maturity of the model-driven approaches both in forward and reverse engineering contexts. Model-driven approaches become available in various application domains with various objectives. Furthermore, there is a trend in all development tools to automate tasks as much as possible to improve productivity and avoid human errors. There are half of the model-driven reverse engineering approaches with semi-automated process, testifying that the automation of tasks requires a significant effort and in the same time it may not be as advanced as not needing a human expert.

VI. Threats to Validity

In this section, we discuss the possible threats of our survey according to the types defined by Wohlin et al. [102]: conclusion, internal, construct, and external.

A. Conclusion Validity

Conclusion validity concerns the reliability of conclusions of the systematic literature survey. A possible threat may result from the fact that not all the relevant studies that exist may be identified. We have tried to limit this aspect by considering the most used and diffused search engines, and during the construction of the applied queries. In the queries, some trade-offs are necessary in the choice of terms and in their composition. For example, looking for "model" and "driven" and "model driven" in the same context (e.g., title, abstract and keywords) yields very different results: in the first case many more works are returned, since all works using both words in unrelated sentences are retrieved, while in the second case we get only papers containing "model driven" but not "model-driven", or "model`<newline>`driven". In our queries, we tried to build different variants of the considered word sequences, but we may have left out some usages that we are not aware of.

B. Internal Validity

Internal validity threats concern the possible errors in the conclusions due to the causal relationship between the research process and the results. Essentially, internal validity indicates how well the findings represent the true opinion expressed in the literature. In this paper, we have followed the systematic research method to limit as much as possible the internal threats.

One threat specific to this paper is the definition of the evaluation framework and the assignment of papers to the different values in the defined features. Many of the reported papers, in fact, define what we consider a MDRE approach without expressing their concepts using the same terminology we apply, and this makes the categorization of the approaches more error-prone. Moreover, the retrieved papers explain some very complex approaches, often describing them partially or superficially (w.r.t. our need of details). For this reason, when we encountered a paper closely related to the considered topic, but not fully classifiable as an MDRE approach, we briefly described it in Appendix A, instead of simply rejecting it. This gives us the chance to discuss it and the reader the knowledge about its relation with the topic.

C. Construct Validity

Construct validity concerns the generalization of the result to the concept or theory after the study execution. This threat is related to the potentially subjective analysis. During this survey, we, the authors, have discussed together all the aspects of a systematic literature survey and verified independently all the information extracted from the primary studies to limit as much as possible this threat. As mentioned in Section II-B, two of the team members have read all the papers provided as results from the search engines. Hence, the paper data has been extracted by two team members. In case of disagreements, the third member of the team has read the paper under investigation and provide a decision. Further, all the team members have met and decided how to classify the paper, i.e., a true positive or a false positive. We also used a framework to reduce the decisions to be taken to a finite set of criteria.

D. External Validity

External validity concerns the ability to generalize the result of the experiment to industrial practice. Essentially, this survey deals with MDRE approaches, most of which have been applied to industrial case studies. Hence, they demonstrate a practical application of the approaches within various industries.

VII. Discussion

We outline below some hints on the criteria which may be considered when choosing an available MDRE approach and some limitations present in the available MDRE approaches.

A. Hints on Choosing a MDRE Approach

Reverse engineering is a complex task and the scientific literature proposes several solutions including the model-driven one. In this paper we reported many MDRE approaches. The question which may raise here is: which one to choose when we have to analyze a legacy system in a given application domain with a specific objective? In the following, we propose four possible hints for this choice.

a) Application domain: A first hint concerns the application domain of the legacy system. If available approaches for that application domain exist, they may be successfully used because they have advantages over general approaches (e.g., there are available domain models, metamodels, tools supporting the reverse engineering process). Theoretically, specific MDRE approaches are specialized and optimized to address meaningful aspects of a given application domain or objective. Several domain specific approaches have been described in Section III:

- Cosentino et al. for business rule extraction from Java software,
- Djamel et al. for addressing semantic Web services,
- El Beggar et al. for COBOL legacy systems,
- Garcia-Rodriguez de Guzman et al. for modernizing legacy databases towards Web services,
- Lenk et al. for 3D Web applications,
- Ristic et al. for legacy relational databases,
- Rodriguez-Echeverria et al. for Web applications using the MVC pattern,
- Sanchez-Ramon et al. for GUI in RAD applications,
- Trias et al. for Web applications,
- Warwas and Klusch for multi-agent systems.

Generic MDRE approaches may be applied to a wide range of application domains and with various purposes of the analysis task. The generic approaches we described in this paper are:

- Bruneliere et al. for various objectives in various application domains,
- Favre et al. for object-oriented software,
- Fleurey et al. for large banking systems,
- Normantas and Vasilecas for enterprise systems,
- Perez-Castillo et al for business processes extraction.

Usually, these generic approaches provide also examples of concrete applications scenarios, hence metamodels, domain models, transformation chains may be already available.

b) Standards: A second hint concerns the standards used by the approach. The use of standards makes possible the exchange of models which can be modified with tools developed by different developers. Obviously, it should be easier to find experts who understand standard models and tools which support standard models. Three of the generic approaches we described, Bruneliere et al., Normantas and Valilecas, and Perez-Castillo et al. exploit KDM models. Djamel et al. use a UML profile, while Favre et al. UML diagrams. Furthermore, a specific approach uses KDM, i.e., Trias et al., while El Beggar et al. and Fleurey et al. use UML concepts.

c) Validation: A third hint regards the possibility to validate the models and how the validation is performed. Most of the approaches do not address this issue. This can be a problem if we do not trust the transformations on models; transformations are usually completely automated, but not error-free. Two of the described approaches deal with model validation. Djamel et al. propose models proof using an ontology reasoner, but the proof is not described by the authors. Favre et al.'s approach to models validation is interesting, because it addresses the consistency proof between models at the different abstraction levels.

d) Tooling: A fourth hint concerns the development/implementation support of the reverse engineering process. As already outlined, all the described MDRE approaches mention the tool support. However, the most well described and spread tool support is provided by Bruneliere et al. through MoDisco, a framework which may ease significantly the reverse engineering effort. Its usefulness is sustained also by the fact that other four different approaches exploit MoDisco for their implementation. We must outline that more tools will become available as confirmed by the most recent tools as MARBLE by Perez-Castillo et al. and RE-CMS by Trias et al., for which we did not discover yet applications outside their authors community.

Considering only the five general purpose MDRE approaches described in this paper, Bruneliere et al. and Favre et al. propose the most interesting and complete approaches for a reverse engineering task in a generic domain. Both meet the second hint (Bruneliere et al. use KDM standards, Favre et al. use UML concepts). Bruneliere et al. meet successfully the fourth hint, while Favre et al. the third.

B. Open Issues for MDRE Approaches

The described papers identify different issues in their approaches, coming from the experience gained during their application. For example, the referenced papers show limitations on: the manipulation of large-scale models, the use of immature development environments, the lack of metamodels aligned to MDRE that support reverse engineering of different languages and the lack of predefined solutions in the static analysis stage that can be applied to a large number of applications.

We discuss below some of these issues that provide useful hints for future investigations and research developments.

A first issue is related to the scalability of the applied technologies: Bruneliere et al. [36], for example, report that the EMF framework they rely on has poor scalability properties by default. More recently, with the advent of dedicated storage layers like CDO, the use of EMF to handle large models has become more feasible.

Another interesting issue raised by some authors is about the traceability of the performed transformations [45], [48]. Lenk et al. [48] addressed the need of tracing equivalent entities in different models by assigning them synthetic IDs. They refer to this solution as "model pollution", since this annotation is part of the model itself. Fleurey et al. [45] instead point to the missing migration of tests, and in particular to the fact that more work should be done to assess what/how to test the newly generated code, and if these new tests cover the use cases defined in the original system. The described problem is mostly related to model transformation traceability, since, e.g., tracing the path of different entities or concepts in different models is useful to fill the target model with missing information, like tests in this case. Traceability in MDE is still an active research area [103].

Another common issue is the role of UML in reverse engineering, which is not well-defined yet. Normantas and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2733518, IEEE Access

Vasilecas [52] refer to the need to provide UML diagrams (and other more specific formats to report business rules) to better communicate the extracted information to developers. Sun et al. [69] instead plan to create a UML profile because UML cannot represent all the needed information. Given the work of OMG on ADM and its metamodels, UML is not probably the best model to be used to support an MDRE process but, as reported above, there is the need to be able to link UML models to the more technical ones used during a discovery process. This is needed to ease the inspection and comprehension of results by development teams, since this is one of the purposes of UML.

With respect to the approach to be used when performing the modeling phase, and the choice of the metamodel to apply, we observed opposite opinions in different authors. For example, Trias et al. [71] assert that the use of ASTM saved them time and effort, and make the hypothesis that its low use is due to its novelty [70] (the specification dates back in 2011, two years before the publication). Warwas and Klusch [75] instead suggest the use of a bottom-up approach, where the metamodels should be near the way the analyzed project is defined (Java agent systems in their case). This is actually the opposite direction than the ADM's one, since the model defined by ADM tend to be heavily standardized, with the capability of being adapted/extended to support additional features. A bottom-up approach, instead, advocates the use of specific models and their integration through lightweight models (like views) that link together different heterogeneous entities.

Finally, the most recurrent discussion [48], [55], [64], [65], [67], [84] about future work in the described papers is related to the extension to different languages or technologies. While this is a natural consideration to be made by authors, it has particular value when dealing with MDE. As we reported in the paper, in fact, the number of case studies reported in the articles are limited, if we consider that the whole MDE is targeted at the reuse of models and processes. In this direction, we argue that more research efforts should be made in this area, allowing to assess the value of each methodology in more than one context.

VIII. Conclusions

In this paper, we propose a systematic literature review of works having the intent to adopt a model-driven reverse engineering (MDRE) approach. We also mentioned other approaches that fit only partially our inclusion criteria, regarding the steps we have identified as characterizing MDRE. Our SLR provides an overview of the currently available approaches and on their main characteristics. We are not aware of any other SLRs on MDRE approaches, at the best of our knowledge.

From our SLR, we observe that there are some research groups which are interested in this field and which continue to publish their results during the last years. We mention Bruneliere et al., Favre et al., Perez-Castillo el at., Sanchez-Ramon et al., Rodriguez-Echeverria et al., and Trias et al. All these cited groups are located in Latin countries mainly in Europe (France and Spain) and South-America (Argentina). There are also some authors who publish spot articles in this field such as Djamel et al, Fleurey et al and Warwas and Klusch.

We also observe that the model-driven reverse engineering field is young: the first papers dates back in 2003–2005 (Qiao et al., Rugaber and Stirewalt, Bouillon et al). Actually, all the identified primary studies have been published in the last ten years. The MDRE approaches are heterogeneous, have various objectives, and are applied in different application domains. The research in this field is growing and we expect to assist to its further development in the next future.

Appendix A
Approaches Partially Described by their Authors

Some MDRE approaches are not completely described by their authors (e.g., information about used metamodels or model transformation is missing), and we briefly introduce them in the following (the approaches indicated in the Cited column in Table II).

Arevalo et al. [27] extends the Business Process Model and Notation (BPMN) metamodel to address time-related issues to handle business temporal rules. The paper focuses mainly on the definition of the metamodel. To evaluate their proposal, the authors present a case study concerning the reverse engineering of a legacy database, where the source system is represented by a MS Project Server and the target system is a BPMN. In their description, the authors provide minimal details concerning the model-driven reverse engineering process. No transformation rules are described.

Bellucci et al. [30] implement a tool for reverse engineering of dynamic Web applications. The objective of the tool is to extract the information available in the source artifacts and to describe them in a model-based way at various abstraction levels, in order to further adapt Web applications to various types of interactive devices. The tool uses two abstraction levels: a platform independent and a platform dependent. However, the authors do not provide details on the used metamodels and on the transformation rules between various models.

F. Barbier et al. [28] provides a model-driven based reverse engineering approach for COBOL legacy applications. The authors have also published a book in 2015 on this topic [104]. Both references are available only through payment.

Bergmayr et al. [32] focus on the dynamic analysis of Java systems. They propose an open framework for reverse engineering of executable software called fREX. Essentially, fREX extracts behavioral aspects as activity diagrams from Java executables and describes them through the OMG's fUML standard language. The approach implements two main steps: the first creates an initial model automatically by using the MoDisco discoverer component for Java, while the second performs a Java-to-UML mapping using an ATL model-to-model transformation language and tooling. The authors does not provide further details on the implementation of the approach. They describe a toy example of a Java class from which it is extracted the class diagram and the activity diagram describing the constructor of the class.

Damasevicius et al. [38] present a framework for the automated derivation of features models from source artifacts. The authors focus on the formal description of the features models, a program-feature relation metamodel, and a method to generate feature models. The reverse engineering process is not entirely metamodel driven, the authors specifying a metamodel only for the target model.

Garzón et al. [46], [105] propose a reverse engineering solution in which models are merged with the code, hence models are represented directly into code. More specifically, the approach is called *umplification*, which means that the source code maintains its behavior while it is enhanced with model-level abstractions, i.e., UML modeling notations. In this way, the distinction between code and models is overcome. No code discoverers or transformation rules are described.

Martinez-Perez et al. [51] propose a reverse engineering approach to extract security policies implemented in network firewalls with the objective to ease the understanding, analysis, and evolution of network security policies. The aim of this approach is to produce a model that abstracts the information available in low-level firewall configurations and that represents this information in terms of hosts, services, and permissions relevant for the global access control policy of a network. This approach is driven by the network access-control metamodel defined by Salvador Martinez-Perez et al. The metamodel captures rules and exceptions concerning network security policies. The authors focus mainly on the definition of the metamodel and provide limited information about the models obtained during the reverse engineering steps.

Ovchinnikova and Asnina propose an approach for software development based on MDE principles called Topological Functional Modeling for Model Driven Architecture (TFM4MDA) [54]. They mention that their approach considers also a reverse engineering part for the migration of a legacy system to a new platform or for the integration of a legacy system with other software systems. Mainly, this approach extracts UML class for capturing the statical aspects and sequence diagrams for dynamical aspects from the source artifacts. Essentially, the authors identify which available tools may be used for implementing the reverse engineering steps for extracting the UML diagrams and describing them in TFM4MDA [106]. The authors focus on the mappings between the UML diagrams and the topological functional modeling and the information about the model-driven reverse engineering process is limited.

Pu et al. [60] propose an approach to reverse engineer Web-based legacy systems with the integration of model-driven engineering and UML. The authors mention how legacy Web applications can be described through UML class, component, and deployment diagrams. Further, the authors provide a tool which implements this approach: SEASAT (Software Evolution for domAin-Specific legAcy sysTems). No details on code discoverers and model-to-model transformation are described. The authors focus on code-to-model mappings.

Qiao et al. [61] introduce their round-trip approach to evolve Web legacy systems towards a model-driven approach. To achieve their objectives, the authors identify the components of a legacy systems through decomposition techniques, components which are further described into the Reengineering Wide Spectrum Language (RWSL) terms. Abstraction rules are then applied on the RWSL code to extract the architectural artifacts and to describe the architecture as components interconnected among them. Finally, the obtained architecture is mapped to an UML profile, which represent the PIM. The authors provide minimum details on the reverse engineering process and on the UML profile.

Rugaber and Stirewalt [8] propose the reverse of the reverse engineering process as a mechanism to evaluate the quality of the reverse engineering results. They illustrate their solution through a numerical application written in the C programming language. They define the application domain and the program model. However, code discoverers and transformation rules are not dealt in this approach.

References

[1] J. Bézivin, "On the unification power of models," *Software & Systems Modeling*, vol. 4, no. 2, pp. 171–188, 5 2005.

[2] Object Management Group, Inc., "Object management group," 2012. [Online]. Available: http://www.omg.org

[3] O. Group, "Mda guide revision 2.0," July 2014. [Online]. Available: http://www.omg.org/cgi-bin/doc?ormsc/14-06-01

[4] A. van Deursen, E. Visser, and J. Warmer, "Model-driven software evolution: A research agenda," in *CSMR Workshop on Model-Driven Software Evolution (MoDSE 2007)*, D. Tamzalit, Ed., Amsterdam, The Netherlands, 3 2007, pp. 41–49. [Online]. Available: http://swerl.tudelft.nl/twiki/pub/Main/TechnicalReports/TUD-SERG-2007-006.pdf

[5] P. Mohagheghi, W. Gilani, A. Stefanescu, M. A. Fernández, B. Nordmoen, and M. Fritzsche, "Where does model-driven engineering help? experiences from three industrial cases," *Software and System Modeling*, vol. 12, no. 3, pp. 619–639, 2013. [Online]. Available: http://dx.doi.org/10.1007/s10270-011-0219-7

[6] V. Amaral and M. Mernik, "Special issue on quality in model-driven engineering," *Software Quality Journal*, vol. 24, no. 3, pp. 597–599, 2016. [Online]. Available: http://dx.doi.org/10.1007/s11219-016-9327-5

[7] O. Pastor, *Model-Driven Development in Practice: From Requirements to Code*. Cham: Springer International Publishing, 2017, pp. 405–410. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-51963-0_31

[8] S. Rugaber and K. Stirewalt, "Model-driven reverse engineering," *IEEE Software*, vol. 21, no. 4, pp. 45–53, 8 2004.

[9] E. G. Scalise P., J.-M. Favre, and N. Zambrano, "Model-driven reverse engineering and program comprehension: an example," *Ingeniare. Revista chilena de ingenier?a*, vol. 18, no. 1, pp. 76–83, 4 2010. [Online]. Available: http://www.scielo.cl/pdf/ingeniare/v18n1/art09.pdf

[10] J.-M. Favre, "Foundations of model (driven) (reverse) engineering – episode I: Story of the fidus papyrus and the solarus," in *Language Engineering for Model-Driven Software Development*, ser. Dagstuhl Seminar Proceedings, J. Bezivin and R. Heckel, Eds., no. 04101. Dagstuhl, Germany: Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany, 2005. [Online]. Available: http://drops.dagstuhl.de/opus/volltexte/2005/13

[11] R. C. Holt, A. Winter, and A. Schurr, "GXL: toward a standard exchange format," in *Proceedings of the Seventh Working Conference on Reverse Engineering*, ser. WCRE 2000. Brisbane, QLD, Australia: IEEE Computer Society, November 2000, pp. 162–171.

[12] H. A. Mï¿½ller, M. A. Orgun, S. R. Tilley, and J. S. Uhl, "A reverse-engineering approach to subsystem structure identification," *Journal of Software Maintenance: Research and Practice*, vol. 5, no. 4, pp. 181–204, 1993. [Online]. Available: http://dx.doi.org/10.1002/smr.4360050402

[13] R. Kazman, L. O'Brien, and C. Verhoef, "Architecture reconstruction guidelines," Software Engineering Institute, Carnegie Mellon University, Tech. Rep. CMU/SEI-2002-TR-034, Nov. 2003. [Online]. Available: http://resources.sei.cmu.edu/library/asset-view.cfm?assetid=6255

[14] O. Nierstrasz, S. Ducasse, and T. Gîrba, "The story of moose: an agile reengineering environment," in *Proceedings of the 10th European software engineering conference held jointly with 13th ACM SIGSOFT international symposium on Foundations of software engineering*, ser. ESEC/FSE-13. Lisbon, Portugal: ACM, September 2005, pp. 1–10.

[15] M. Zanoni, F. Arcelli Fontana, and F. Stella, "On applying machine learning techniques for design pattern detection," *Journal of Systems and Software*, vol. 103, pp. 102–117, 2015. [Online]. Available: http://dx.doi.org/10.1016/j.jss.2015.01.037

[16] B. Kitchenham, "Procedures for performing systematic reviews," Keele University, Tech. Rep., Jul. 2004.

[17] P. Brereton, B. A. Kitchenham, D. Budgen, M. Turner, and M. Khalil, "Lessons from applying the systematic literature review process within the software engineering domain," *Journal of Systems and Software*, vol. 80, no. 4, pp. 571–583, 2007, software Performance - 5th International Workshop on Software and Performance.

[18] B. A. Kitchenham, D. Budgen, and O. P. Brereton, "Using mapping studies as the basis for further research - A participant-observer case study," *Information & Software Technology*, vol. 53, no. 6, pp. 638–651, 2011.

[19] F. Arcelli Fontana, R. Roveda, M. Zanoni, C. Raibulet, and R. Capilla, "An experience report on detecting and repairing software architecture erosion," in *13th Working IEEE/IFIP Conference on Software Architecture, WICSA 2016, Venice, Italy, April 5-8, 2016*, 2016, pp. 21–30.

[20] F. Arcelli Fontana, S. Masiero, and C. Raibulet, "Elemental design patterns recognition in java," in *13th International Workshop on Software Technology and Engineering Practice (STEP 2005), 24-25 September 2005, Budapest, Hungary*, 2005, pp. 196–205.

[21] F. Arcelli Fontana, S. Maggioni, and C. Raibulet, "Design patterns: a survey on their micro-structures," *Journal of Software: Evolution and Process*, vol. 25, no. 1, pp. 27–52, 2013. [Online]. Available: http://dx.doi.org/10.1002/smr.547

[22] F. Arcelli Fontana, F. Perin, C. Raibulet, and S. Ravani, "JADEPT: dynamic analysis for behavioral design pattern detection," in *ENASE 2009 - Proceedings of the 4th International Conference on Evaluation of Novel Approaches to Software Engineering, Milan, Italy, May 2009*, 2009, pp. 95–106.

[23] F. Arcelli Fontana, S. Maggioni, and C. Raibulet, "Understanding the relevance of micro-structures for design patterns detection," *Journal of Systems and Software*, vol. 84, no. 12, pp. 2334–2347, 2011. [Online]. Available: http://dx.doi.org/10.1016/j.jss.2011.07.006

[24] F. Arcelli Fontana, S. Masiero, C. Raibulet, and F. Tisato, "A comparison of reverse engineering tools based on design pattern decomposition," in *16th Australian Software Engineering Conference (ASWEC 2005), 31 March - 1 April 2005, Brisbane, Australia*. IEEE Computer Society, 2005, pp. 262–269. [Online]. Available: http://dx.doi.org/10.1109/ASWEC.2005.5

[25] F. Arcelli Fontana, C. Raibulet, and M. Zanoni, "Alternatives to the knowledge discovery metamodel: An investigation," *International Journal of Software Engineering and Knowledge Engineering*, vol. 27, no. 7, pp. 1–32, 2017.

[26] F. Amendola and L. Favre, "Science & engineering software migration: Moving from desktop to mobile applications," in *Proceedings of the 5th International Conference on Computational Methods for Coupled Problems in Science end Engineering, Ibiza, Spain*, 2013, pp. 1418–1429.

[27] C. Arevalo, M. J. E. Cuaresma, I. M. Ramos, and M. Domínguez-Muñoz, "A metamodel to integrate business processes time perspective in BPMN 2.0," *Information & Software Technology*, vol. 77n, pp. 17–33, 2016. [Online]. Available: http://dx.doi.org/10.1016/j.infsof.2016.05.004

[28] F. Barbier, S. Eveillard, K. Youbi, O. Guitton, A. Perrier, and E. Cariou, "Chapter 11 - model-driven reverse engineering of cobol-based applications," in *Information Systems Transformation*, ser. The MK/OMG Press, W. M. Ulrich and P. H. Newcomb, Eds. Boston: Morgan Kaufmann, 2010, pp. 283–299.

[29] G. Barbier, H. Bruneliere, F. Jouault, Y. Lennon, and F. Madiot, "Chapter 14 - modisco, a model-driven platform to support real legacy modernization use cases," in *Information Systems Transformation*, ser. The MK/OMG Press, W. M. Ulrich and P. H. Newcomb, Eds. Boston: Morgan Kaufmann, 2010, pp. 365–400.

[30] F. Bellucci, G. Ghiani, F. Paternò, and C. Porta, "Automatic reverse engineering of interactive dynamic web applications to support adaptation across platforms," in *17th International Conference on Intelligent User Interfaces, IUI '12, Lisbon, Portugal, February 14-17, 2012*, 2012, pp. 217–226.

[31] D. A. Bensaber and M. Malki, "Development of semantic web services: Model driven approach," in *8th International Conference on New Technologies of Distributed Systems, Lyon, France, June 23-27, 2008*, 2008, pp. 40–51.

[32] A. Bergmayr, H. Brunelière, J. Cabot, J. García, T. Mayerhofer, and M. Wimmer, "frex: fuml-based reverse engineering of executable behavior for software dynamic analysis," in *Proceedings of the 8th International Workshop on Modeling in Software Engineering, MiSE@ICSE 2016, Austin, Texas, USA, May 16-17, 2016*, 2016, pp. 20–26.

[33] A. Bergmayr, H. Bruneliere, J. L. C. Izquierdo, J. Gorronogoitia, G. Kousiouris, D. Kyriazis, P. Langer, A. Menychtas, L. Orue-Echevarria, C. Pezuela, and M. Wimmer, "Migrating legacy software to the cloud with artist," in *17th European Conference on Software Maintenance and Reengineering (CSMR) 2013, Genova, Italy, March 5 -8, 2013*, 2013, pp. 465–468.

[34] C. Blanco, R. Pérez-Castillo, A. Hernández, E. Fernández-Medina, and J. Trujillo, *Data Warehousing and Knowledge Discovery: 11th International Conference, DaWaK 2009 Linz, Austria, August 31–September 2, 2009 Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, ch. Towards a Modernization Process for Secure Data Warehouses, pp. 24–35.

[35] L. Bouillon, Q. Limbourg, J. Vanderdonckt, and B. Michotte, "Reverse engineering of web pages based on derivations and transformations," in *Third Latin American Web Congress (LA-WEB 2005)*. Buenos Aires, Argentina: IEEE, Oct 2005, p. 11.

[36] H. Bruneliï¿$\frac{1}{2}$re, J. Cabot, G. Dupï¿$\frac{1}{2}$, and F. Madiot, "MoDisco: A model driven reverse engineering framework," *Information and Software Technology*, vol. 56, no. 8, pp. 1012–1032, 2014.

[37] V. Cosentino, J. Cabot, P. Albert, P. Bauquel, and J. Perronnet, "Extracting business rules from cobol: A model-based framework," in *20th Working Conference on Reverse Engineering (WCRE 2013)*, Oct 2013, pp. 409–416.

[38] R. Damasevicius, P. Paskevicius, E. Karciauskas, and R. Marcinkevicius, "Automatic extraction of features and generation of feature models from java programs," *ITC*, vol. 41, no. 4, pp. 376–384, 2012. [Online]. Available: http://dx.doi.org/10.5755/j01.itc.41.4.1108

[39] A. B. Djamel, B. Djamal, and M. Mimoun, "A reverse engineering approach for specifying semantic web service with respect to MDA," *International Review on computers and software*, vol. 2, no. 6, pp. 581–593, Nov. 2007. [Online]. Available: http://liris.cnrs.fr/publis/?id=3349

[40] O. El Beggar, B. Bousetta, and T. Gadi, "Comparative study between clustering and model driven reverse engineering approaches," *Lecture Notes on Software Engineering*, vol. 1, no. 2, pp. 135–140, 2013.

[41] L. Favre, "Formalizing MDA-based reverse engineering processes," in *Proceedings of the 6th ACIS International Conference on Software Engineering Research, Management and Applications (SERA 2008)*. Prague, Czech Republic: IEEE Computer Society, 2008, pp. 153–160.

[42] L. Favre, L. Martinez, and C. Pereira, "MDA-based reverse engineering of object oriented code," in *Enterprise, Business-Process and Information Systems Modeling*, ser. Lecture Notes in Business Information Processing, T. Halpin, J. Krogstie, S. Nurcan, E. Proper, R. Schmidt, P. Soffer, and R. Ukor, Eds. Springer Berlin Heidelberg, 2009, vol. 29, pp. 251–263.

[43] L. Favre, L. Martinez, C. Pereira, and C. Bunse, *Emerging Technologies for the Evolution and Maintenance of Software Models*. Hershey: IGI Global, 2012, ch. Software System Modernization: An MDA-Based Approach, pp. 164–199.

[44] L. Favre, L. Martinez, and C. Pereira, *Reverse Engineering of Object-Oriented Code: An ADM Approach*. IGI Global, 2014, vol. anche.

[45] F. Fleurey, E. Breton, B. Baudry, A. Nicolas, and J.-M. J?z?quel, "Model-driven engineering for software migration in a large industrial context," in *Model Driven Engineering Languages and Systems*, ser. Lecture Notes in Computer Science, G. Engels, B. Opdyke, D. Schmidt, and F. Weil, Eds. Springer Berlin Heidelberg, 2007, vol. 4735, pp. 482–497.

[46] M. A. Garzón, T. C. Lethbridge, H. Aljamaan, and O. Badreddin, "Reverse engineering of object-oriented code into umple using an incremental and rule-based approach," in *Proceedings of 24th Annual International Conference on Computer Science and Software Engineering (CASCON '14)*. Markham, Ontario, Canada: IBM Corp., 2014, pp. 91–105.

[47] I. Garcia-Rodriguez de Guzman, M. Polo, and M. Piattini, "An adm approach to reengineer relational databases towards web services," in *Proceedings of the 14th Working Conference on Reverse Engineering (WCRE 2007)*. Vancouver, BC: IEEE, Oct. 2007, pp. 90–99.

[48] M. Lenk, A. Vitzthum, and B. Jung, "Model-driven iterative development of 3d web-applications using ssiml, x3d and javascript,"

in *Proceedings of the 17th International Conference on 3D Web Technology (Web3D '12)*. Los Angeles, California: ACM, Aug. 2012, pp. 161–169.

[49] L. Martinez, C. Pereira, and L. Favre, "Recovering sequence diagrams from object-oriented code - an ADM approach," in *Proceedings of the 9th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE 2014)*, Lisbon, Portugal, April 2014, pp. 188–195.

[50] L. Martinez, L. Favre, and C. Pereira, *Architecture-driven modernization for software reverse engineering technologies*. IGI Global, 2013. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-84902341025&partnerID=40&md5=9e498a81399a0b24eae41a4c2293ce1a

[51] S. Martínez Perez, J. García-Alfaro, F. Cuppens, N. Cuppens-Boulahia, and J. Cabot, "Model-driven extraction and analysis of network security policies," in *Model-Driven Engineering Languages and Systems - 16th International Conference, MODELS 2013, Miami, FL, USA, September 29 - October 4, 2013. Proceedings*, 2013, pp. 52–68.

[52] K. Normantas and O. Vasilecas, "Extracting business rules from existing enterprise software system," in *Information and Software Technologies*, ser. Communications in Computer and Information Science, T. Skersys, R. Butleris, and R. Butkiene, Eds. Springer Berlin Heidelberg, 2012, vol. 319, pp. 482–496.

[53] K. Normantas, S. Sosunovas, and O. Vasilecas, "An overview of the knowledge discovery meta-model," in *Proceedings of the 13th International Conference on Computer Systems and Technologies (CompSysTech '12)*. Ruse, Bulgaria: ACM, 2012, pp. 52–57.

[54] V. Ovchinnikova and E. Asnina, "The algorithm of transformation from UML sequence diagrams to the topological functioning model," in *ENASE 2015 - Proceedings of the 10th International Conference on Evaluation of Novel Approaches to Software Engineering, Barcelona, Spain, 29-30 April, 2015.*, 2015, pp. 377–384.

[55] C. Pereira, L. Martinez, and L. Favre, "Recovering use case diagrams from object oriented code: An MDA-based approach," in *Proceedings of the Eighth International Conference on Information Technology: New Generations (ITNG 2011)*, April 2011, pp. 737–742.

[56] R. Perez-Castillo, M. Fernandez-Ropero, I. Garcia-Rodriguez de Guzman, and M. Piattini, "MARBLE. a business process archeology tool," in *Proceedings of the 27th IEEE International Conference on Software Maintenance (ICSM 2011)*. Williamsburg, VI: IEEE, Sep. 2011, pp. 578–581.

[57] R. Pérez-Castillo, I. Garcìa-Rodrìguez de Guzmán, and M. Piattini, "Knowledge Discovery Metamodel-ISO/IEC 19506: A standard to modernize legacy systems," *Computer Standards & Interfaces*, vol. 33, no. 6, pp. 519–532, 2011.

[58] R. Pérez-Castillo, B. Weber, I. G. R. de Guzmán, and M. Piattini, "Process mining through dynamic analysis for modernising legacy systems," *IET Software*, vol. 5, no. 3, pp. 304–319, 2011. [Online]. Available: http://dx.doi.org/10.1049/iet-sen.2010.0103

[59] M. Polo, I. G. R. de Guzmán, and M. Piattini, "An mda-based approach for database re-engineering," *Journal of Software Maintenance*, vol. 19, no. 6, pp. 383–417, 2007.

[60] J. Pu, H. Yang, B. Xu, L. Xu, and W. C.-C. Chu, "Combining MDE and UML to reverse engineer web-based legacy systems," in *Proceedings of the 32nd Annual IEEE International Computer Software and Applications Conference (COMPSAC '08)*. Turku, Finland: IEEE Computer Society, Jul. 2008, pp. 718–725.

[61] B. Qiao, H. Yang, W. C. Chu, and B. Xu, "Bridging legacy systems to model driven architecture," in *27th International Computer Software and Applications Conference (COMPSAC 2003): Design and Assessment of Trustworthy Software-Based Systems, 3-6 November 2003, Dallas, TX, USA, Proceedings*, 2003, p. 304.

[62] S. Ristic, S. Kordic, M. Celikovic, V. Dimitrieski, and I. Lukovic, "A model-driven approach to data structure conceptualization," in *2015 Federated Conference on Computer Science and Information Systems, FedCSIS 2015, Lódz, Poland, September 13-16, 2015*, 2015, pp. 977–984.

[63] S. Ristic, S. Aleksic, M. Celikovic, and I. Lukovic, "Generic and standard database constraint meta-models," *Comput. Sci. Inf. Syst.*, vol. 11, no. 2, pp. 679–696, 2014.

[64] R. Rodríguez-Echeverría, V. M. Pavón, F. Macías, J. M. Conejero, P. J. Clemente, and F. Sánchez-Figueroa, *Web Information Systems Engineering – WISE 2013: 14th International Conference, Nanjing, China, October 13-15, 2013, Proceedings, Part II*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, ch. Generating a Conceptual Representation of a Legacy Web Application, pp. 231–240.

[65] O. Sánchez Ramón, J. Sánchez Cuadrado, and J. García Molina, "Model-driven reverse engineering of legacy graphical user interfaces," in *Proceedings of the IEEE/ACM International Conference on Automated Software Engineering (ASE '10)*. Antwerp, Belgium: ACM, 2010, pp. 147–150.

[66] O. Ramon, J. Cuadrado, and J. Molina, "Reverse engineering of event handlers of rad-based applications," in *Proceedings of the 18th Working Conference on Reverse Engineering (WCRE 2011)*, Oct 2011, pp. 293–302.

[67] O. Sanchez Ramon, J. Vanderdonckt, and J. Garcia Molina, "Re-engineering graphical user interfaces from their resource files with UsiResourcer," in *Proceedings of the IEEE 7th International Conference on Research Challenges in Information Science (RCIS 2013)*, Paris, France, May 2013, pp. 1–12.

[68] E. Sosa Sánchez, P. J. Clemente, M. Sanchez-Cabrera, J. M. Conejero, R. Rodríguez-Echeverría, and F. Sánchez-Figueroa, "Service discovery using a semantic algorithm in a SOA modernization process from legacy web applications," in *2014 IEEE World Congress on Services, SERVICES 2014, Anchorage, AK, USA, June 27 - July 2, 2014*, 2014, pp. 470–477.

[69] W. Sun, S. Li, D. Zhang, and Y. Yan, "A model-driven reverse engineering approach for semantic web services composition," *World Congress on Software Engineering*, vol. 3, pp. 101–105, 2009.

[70] F. Trias, V. Castro, M. López-Sanz, and E. Marcos, *Evaluation of Novel Approaches to Software Engineering: 8th International Conference, ENASE 2013, Angers, France, July 4-6, 2013, Revised Selected Papers*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, ch. Reverse Engineering Applied to CMS-Based Web Applications Coded in PHP: A Proposal of Migration, pp. 241–256.

[71] F. Trias, V. de Castro, M. L?pez-Sanz, and E. Marcos, "An adm-based method for migrating cms-based web applications: Extracting astm models from php code," in *Proceedings of the 1st International Workshop in Software Evolution and Modernization (SEM 2013)*. Angers, France: SCITEPRESS, Jul. 2013, pp. 85–92.

[72] F. Trias, V. de Castro, M. L.ez-Sanz, and E. Marcos, "An adm-based method for migrating cms-based web applications," in *25th International Conference on Software Engineering and Knowledge Engineering (SEKE 2013)*. Boston, USA: Knowledge Systems Institute, Jun. 2013, pp. 256–261.

[73] F. Trias, V. de Castro, M. Lopez-Sanz, and E. Marcos, "Migrating traditional web applications to cms-based web applications," *Electronic Notes in Theoretical Computer Science*, vol. 314, pp. 23–44, 2015, {CLEI} 2014, the {XL} Latin American Conference in Informatic.

[74] F. Trias, V. de Castro, M. L. Sanz, and E. Marcos, "RE-CMS: a reverse engineering toolkit for the migration to cms-based web applications," in *Proceedings of the 30th Annual ACM Symposium on Applied Computing, Salamanca, Spain, April 13-17, 2015*, 2015, pp. 810–812.

[75] S. Warwas and M. Klusch, "Making multiagent system designs reusable: A model-driven approach," in *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT 2011)*, vol. 2. Lyon, France: IEEE, Aug. 2011, pp. 101–108.

[76] Object Management Group, Inc., "Architecture-driven modernization task force," Web site, 2012. [Online]. Available: http://adm.omg.org/

[77] P. Newcomb, "Architecture-driven modernization (ADM)," in *Proceedings of the 12th Working Conference on Reverse Engineering (WCRE '05)*. Pittsburgh, Pennsylvania, USA: IEEE Computer Society, November 2005, p. 237.

[78] Object Management Group, Inc., "Knowledge discovery metamodel (kdm)," Web page, 2012. [Online]. Available: http://www.omg.org/technology/kdm/index.htm

[79] *Architecture-driven Modernization: Abstract Syntax Tree Metamodel (ASTM)*, OMG, Jan. 2011, OMG Document Number: formal/2011-01-05. [Online]. Available: http://www.omg.org/spec/ASTM

[80] *Structured Metrics Meta-model (SMM)*, v1.1.1 ed., OMG, Apr. 2016, OMG Document Number: formal/2016-04-04. [Online]. Available: http://www.omg.org/spec/SMM/

[81] *Structured Patterns Metamodel Standard (SPMS)*, OMG, Mar. 2016, OMG Document Number: ptc/16-03-13. [Online]. Available: http://www.omg.org/spec/SPMS/1.1

[82] *Automated Source Code Maintainability Measure™ (ASCMM™)*, v1.0 ed., OMG, Jan. 2016, OMG Document Number: formal/2016-01-01. [Online]. Available: http://www.omg.org/spec/ASCMM/1.0

[83] Object Management Group, Inc., "Meta Object Facility (MOF) core specification," January 2006. [Online]. Available: http://www.omg.org/spec/MOF/2.0/PDF/

[84] V. Cosentino, J. Cabot, P. Albert, P. Bauquel, and J. Perronnet, "A model driven reverse engineering framework for extracting business rules out of a java application," in *Rules on the Web: Research and Applications*, ser. Lecture Notes in Computer Science, A. Bikakis and A. Giurca, Eds. Springer Berlin Heidelberg, 2012, vol. 7438, pp. 17–31.

[85] D. Đurić, "MDA-based ontology infrastructure," *Computer Science and Information Systems*, vol. 1, no. 1, pp. 91–116, 2004.

[86] R. Grønmo, M. Jaeger, and H. Hoff, "Transformations between UML and OWL-S," in *Model Driven Architecture - Foundations and Applications*, ser. Lecture Notes in Computer Science, A. Hartman and D. Kreische, Eds. Springer Berlin Heidelberg, 2005, vol. 3748, pp. 269–283.

[87] L. Martinez, C. Pereira, and L. Favre, "Reverse engineering activity diagram from object-oriented code: An MDA-based approach," *Computer Technology and Application*, vol. 2, no. 12, pp. 969–978, Dec 2011.

[88] P. Tonella and A. Potrich, *Reverse Engineering of Object Oriented Code*, ser. Monographs in Computer Science. Springer, 2005.

[89] R. Pérez-Castillo, I. Garcìa-Rodréguez de Guzmán, I. Caballero, and M. Piattini, "Software modernization by recovering web services from legacy databases," *Journal of Software: Evolution and Process*, vol. 25, no. 5, pp. 507–533, 2013.

[90] R. Perez-Castillo, I. Garc?a-Rodr?guez de Guzman, and M. Piattini, "On the use of ADM to contextualize data on legacy source code for software modernization," in *Proceedings of the 16th Working Conference on Reverse Engineering (WCRE '09)*, Lille, France, Oct 2009, pp. 128–132.

[91] B. Jung, M. Lenk, and A. Vitzthum, "Structured development of 3d applications: round-trip engineering in interdisciplinary teams," *Computer Science - Research and Development*, vol. 30, no. 3, pp. 285–301, 2014.

[92] R. Pérez-Castillo, I. G. R. de Guzmán, and M. Piattini, "Business process archeology using MARBLE," *Information & Software Technology*, vol. 53, no. 10, pp. 1023–1044, 2011.

[93] R. Pérez-Castillo, "MARBLE: modernization approach for recovering business processes from legacy information systems," in *28th IEEE International Conference on Software Maintenance, ICSM 2012, Trento, Italy, September 23-28, 2012*, 2012, pp. 671–676.

[94] E. Sosa, P. Clemente, J. Conejero, and R. Rodriguez-Echeverria, "A model-driven process to modernize legacy web applications based on service oriented architectures," in *Proceedings of the 15th IEEE International Symposium on Web Systems Evolution (WSE 2013)*, Eindhoven, The Netherlands, Sept 2013, pp. 61–70.

[95] R. Rodríguez-Echeverría, J. Conejero, P. Clemente, M. Villalobos, and F. Sanchez-Figueroa, "Generation of webml hypertext models from legacy web applications," in *Proceedings of the 14th IEEE International Symposium on Web Systems Evolution (WSE 2012)*. Trento, Italy: IEEE, Sep. 2012, pp. 91–95.

[96] R. Rodríguez-Echeverría, J. M. Conejero, P. J. Clemente, M. D. Villalobos, and F. Sánchez-Figueroa, *Web Engineering: 12th International Conference, ICWE 2012, Berlin, Germany, July 23-27, 2012. Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, ch. Extracting Navigational Models from Struts-Based Web Applications, pp. 419–426.

[97] R. Rodríguez-Echeverría, F. Macías, V. M. Pavón, J. M. Conejero, and F. Sánchez-Figueroa, *Current Trends in Web Engineering: ICWE 2013 International Workshops ComposableWeb, QWE, MDWE, DMSSW, EMotions, CSE, SSN, and PhD Symposium, Aalborg, Denmark, July 8-12, 2013. Revised Selected Papers*. Cham: Springer International Publishing, 2013, ch. Model-Driven Generation of a REST API from a Legacy Web Application, pp. 133–147.

[98] R. Rodríguez-Echeverría, J. M. Conejero, P. J. Clemente, J. C. Preciado, and F. Sánchez-Figueroa, *Current Trends in Web Engineering: Workshops, Doctoral Symposium, and Tutorials, Held at ICWE 2011, Paphos, Cyprus, June 20-21, 2011. Revised Selected Papers*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, ch. Modernization of Legacy Web Applications into Rich Internet Applications, pp. 236–250.

[99] Q. Limbourg and J. Vanderdonckt, "USIXML: A user interface description language supporting multiple levels of independence," in *Engineering Advanced Web Applications: Proceedings of Workshops in connection with the 4th International Conference on Web Engineering (ICWE 2004)*. Munich, Germany: Rinton press, July 2004, pp. 325–338. [Online]. Available: http://www.usixml.org/servlet/Repository/limbourg-jwe2004.pdf?ID=1513

[100] J. Vanderdonckt, "A MDA-compliant environment for developing user interfaces of information systems," in *Advanced Information Systems Engineering*, ser. Lecture Notes in Computer Science, O. Pastor and J. Falc?o e Cunha, Eds. Springer Berlin Heidelberg, 2005, vol. 3520, pp. 16–31.

[101] A. Benelallam, A. Gómez, G. Sunyé, M. Tisi, and D. Launay, *Neo4EMF, A Scalable Persistence Layer for EMF Models*. Cham: Springer International Publishing, 2014, pp. 230–241.

[102] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in Software Engineering*. Springer Berlin Heidelberg, 2012.

[103] J. M. Vara, V. A. Bollati, . Jimï¿$\frac{1}{2}$nez, and E. Marcos, "Dealing with traceability in the MDD of model transformations," *IEEE Transactions on Software Engineering*, vol. 40, no. 6, pp. 555–583, Jun. 2014.

[104] F. Barbier and J.-L. Recoussine, *Cobol Software Modernization*. John Wiley & Sons, Inc., Jan. 2015.

[105] M. Garzón, H. I. Aljamaan, and T. C. Lethbridge, "Umple: A framework for model driven development of object-oriented systems," in *22nd IEEE International Conference on Software Analysis, Evolution, and Reengineering, SANER 2015, Montreal, QC, Canada, March 2-6, 2015*, 2015, pp. 494–498.

[106] V. Ovchinnikova and E. Asnina, "Overview of the software tools for obtaining uml class diagrams and sequence diagrams from source code within tfm4mda," *Baltic Journal of Modern Computing*, vol. 2, no. 4, pp. 260–271, 2014.