Journal of Web Engineering, Vol. 17, No.3&4 (2018) 183-205

CHALLENGES FOR THE ADOPTION OF MODEL-DRIVEN WEB ENGINEERING APPROACHES IN INDUSTRY

ESTEBAN ROBLES LUNA
Research and Training in Advanced Computing Laboratory (LIFIA), National University of La Plata, La Plata (Argentina)
esteban.robles@lifia.info.unlp.edu.ar

JUAN MIGUEL SÁNCHEZ BEGINES
Web Engineering and Early Testing Group (IWT2), University of Seville, Seville (Spain)
juan.sanchez@iwt2.org

JOSÉ MATÍAS RIVERO
Research and Training in Advanced Computing Laboratory (LIFIA), National University of La Plata, La Plata (Argentina)
mrivero@lifia.info.unlp.edu.ar

LETICIA MORALES
Web Engineering and Early Testing Group (IWT2), University of Seville, Seville (Spain)
leticia.morales@iwt2.org

J.G. ENRÍQUEZ
Computer Languages and Systems Department. University of Seville. Av. Reina Mercedes s/n, 41012, Seville, Seville.
jose.gonzalez@iwt2.org

GUSTAVO ROSSI
Research and Training in Advanced Computing Laboratory (LIFIA), National University of, La Plata, La Plata (Argentina)
gustavo@lifia.info.unlp.edu.ar

Received July 27, 2017
Revised April 23, 2018

Model-Driven Web Engineering approaches have become an attractive research and technology solution for Web application development. However, for more than 20 years of development, the industry has not adopted them due to the mismatch between technical versus research requirements. In the context of this joint work between academia and industry, the authors conduct a survey among hundreds of engineers from different companies around the world and, by statistical analysis, they present the current problems of these approaches in scale. Then, a set of guidelines is provided to improve Model-Driven Web Engineering approaches in order to make them viable industry solutions.

Key words: Model-Driven Web Engineering, Human-Centered interfaces, MDE, Industry

1 Introduction

Model-Driven Web Engineering (MDWE) approaches appeared 20 years ago to fulfill a missing area of Model-Driven Development: Web Application Development [1]. From that moment, about 7 to 8 MDWE solutions [2] have been created, but only a few of them have ended up providing tool support. Also, only one has become the mayor player with small company and support from Object Management Group (OMG) to convert its language into a standard.

Although MDWEs approaches have claimed to improve multiple aspects of Web Application Development such as code quality, development speed and level of abstraction, we have formulated the following main research question: Why has the industry paid little attention to MDWE?

A recent study [3] has presented the 20 obstacles that disturb Web application scalability and, though they are not specifically targeted to the applications derived from MDWE approaches, they are still affected by these applications.

This study presents a list of issues that hinders the use of MDWE approaches in medium to big size companies. As a consequence, it shows precise practical problems that need to be solved to support the claims that the MDWE community has made for years. Figure 1 presents an agile development lifecycle process of a Web application in order to provide a context for the issues that have been found. In this figure, the main phases appear in bold type format. Similarly, the issues that complicate the usage of MDWE approach are represented by "post-it notes" linked to the phase in which each of them have been located [4].

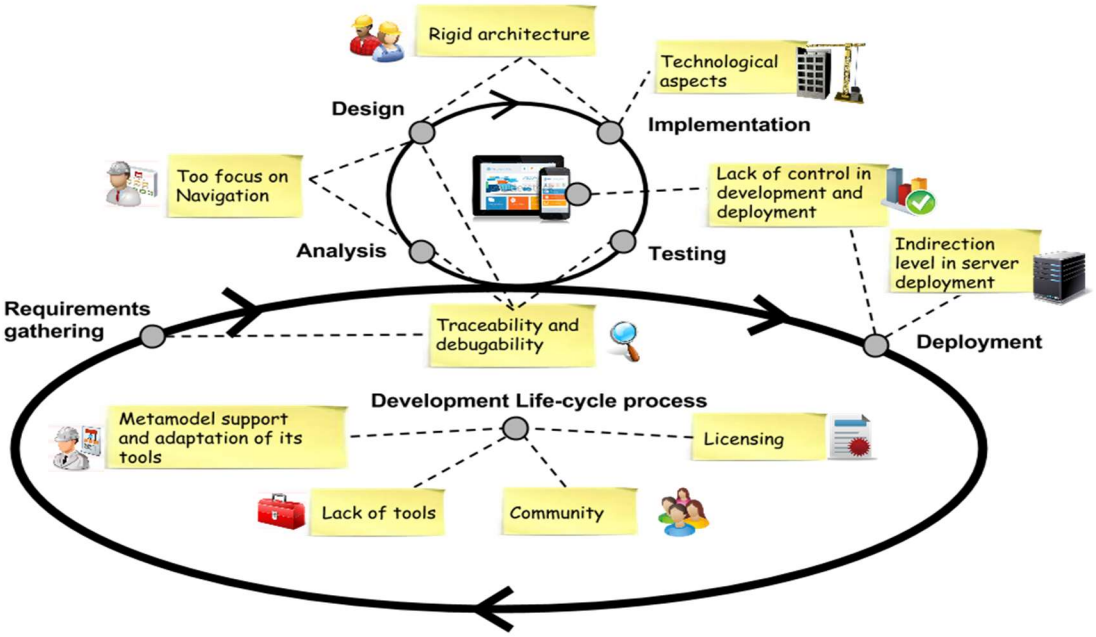

Figure 1 Issues that hinder the usage of the MDWE approach in medium to big size companies

The study uses small experiments to run a MDWE approach in a company, in addition to an exhaustive literature review to give support to its needs. As many different aspects are considered (not only technical) when dealing with limitations of MDWE approaches, we have created a categorization to stress the area where the issue is located:

- Social [S]: A social issue is related to problems among people involved in the project or in the software artifact.
- Technical [T]: A technical issue is associated with the software elements that constitute the Web application.
- Economical [E]: An economical issue is linked to the project's budget or expenses that the development of the application entails.

Table 1 shows the list of issues under these three categories that will be presented in the following subsections. It has been obtained from the interviews we conducted with the engineers. We have decided to include not only technical aspects (as all MDWE approaches take into account), but also social and economic ones. Modern Web Engineering approaches in industry are affected for at least 3 of them. Each subsection explains the concerns and offers a final guideline to solve them.

	Social	Technical	Economical
Lack of control in development and deployment	X	X	X
Too focused in navigation		X	
Metamodel support and adaptation of its tools	X	X	
Traceability and debugability		X	
Lack of tools		X	
Rigid architecture		X	
Technological aspects		X	
Community	X		X
Licensing			X

Table 1. List of issues by category

The experiment is based on a survey that evaluates both the characteristics of the companies participating in the project and the guide to solve the identified problems. This survey tries to address all the issues from the social, technical and economic point of view.

The aim of the survey is to extract helpful information from these companies in order to validate or refute our hypotheses in some way. The results obtained will be considered reliable, since all the companies that have participated in this experiment are of a relevant size and reputation in the software world. Each of the companies has provided specialists in the field such as project managers, analysts or senior programmers who are familiar with the technical problems they tackle.

After the execution of the experiment, we will state whether the hypotheses are valid, that is, whether they solve the problem of each of the phases of Web development, or in contrast, it is necessary to reformulate a new solution guide. This can open the field of MDWE approaches to achieve a framework or standard that may cover the needs that companies demand from this methodology and this way, the industry will pay more attention to them.

In this paper, we will first introduce the MDWE approach and its problems to be used. In section 2, we will present the work related to these problems. In section 3, we will describe the experimentation phase and in section 4 we will propose a series of challenges to solve some of the problems identified

together with a survey. Next, data obtained in the survey will be analyzed and validated in sections 5 and 6, respectively. To end up, section 7 will state a set of conclusions and it will advance future lines of work.

2 Related Work

Since the beginning of the 21st century, the software community has widely accepted modeling languages for Web Applications Development. The MDWE approach has tried to incorporate these languages into the whole development lifecycle in their several stages.

From the onset of the MDWE approach, several solutions have emerged that have attempted to support different Web technologies and architectures (such as Rich Internet Application (RIA), Semantic Web or Mobile Web, among others) and have tried to improve the coverage of the whole development lifecycle of applications according to the specification of requirements, testing and maintenance [5].

Various solutions have been proposed concerning the negative aspect of having a myriad of design approaches and annotations in the MDWE. The first one is to unify the different methods by extracting the best parts from each of them [7]. The second proposes to make all approaches interoperate in one way or another by allowing better portability among design models and facilitating their connections [8]. The feasibility of creating a standard has been studied and finally the OMG has adopted the Interaction Flow Modeling Language (IFML) standard as a solution to that problem.

Another important aspect that the MDWE approach uncovers is to provide support tools and techniques that would play an important role in simplifying the design, development and evolution phases of applications. Computer-Aided Software Engineering (CASE) tools play a key role for software engineers to increase productivity. Therefore, these tools are very significant when adopting the MDWE approach, as they will have a great impact on the improvement of software industry that feeds on such tools [5].

The impact of MDWE approaches on industry has not been sufficiently studied, although we can argue that the application of Model-Driven Software Engineering (MDSE) is widely adopted in various domains and there is a large number of tools that have already assimilated this paradigm.

MDSE bases its development on specific domain models rather than on more general and standardized languages like Unified Modeling Language (UML) [9,10]. There are many negative aspects found in companies when adopting MDSE as, for instance, the effort made in training people to use MDSE techniques. As we will analyze later, tools not supporting the paradigm properly, among other aspects such as immaturity, complexity and usability, provoke increases in costs when applying MDSE methodologies [11].

Another point of debate will be the need to acquire proprietary software or otherwise to create a large community that develops free tools to support MDWE. Eclipse Modeling Tools suite is the premier open source technology for deploying MDE used by IBM and Oracle. There are other applications, like JetBrains Meta Programming System, that are proprietary technologies. In contrast, tools can be different depending on the modeling language: some of them use UML language and some others a specific one.

Finally, it is worth mentioning the effort that academic communities have made in order to develop tools to support MDWE paradigms such as NDT-Suite [12], QuEF [13] and WebRatio [14]. NDT-Suite, for instance, was submitted to a survey in different companies where it obtained satisfactory results in terms of functionality, usability and utility [15].

Another issue that we will address in the survey is the importance of having a broad community that assists and compiles information of interest for a better implementation of the MDWE paradigm. The

MDWE community grew at a rapid pace thanks to the first Web Engineering workshop held at the World Wide Web (WWW) conference in 1998 that focused on Web-based modeling techniques. This enhanced a greater understanding for the community of the problems involved in the development driven by Web application models, thus forming a large volume of knowledge that was published later in different places. This laid an excellent foundation in the MDWE paradigm for professionals in the area [5].

All these MDWE issues will be addressed in this paper from the perspective of users in companies that will evaluate them according to their criteria. A series of solutions or improvements related to the aforementioned arguments will be proposed and validated based on a survey carried out by users.

3 Experimentation

It is necessary to know the points of view of different profiles related to the object of analysis so that the MDWE approach can be evaluated. That is why we have conducted this study through a survey.

In Software Engineering (SE), surveys are one of the most widely used research methods to carry out empirical research studies. The SE research differs from other disciplines, particularly with respect to its objectives and topics. Objectives are often associated either with improving the development process or exploring new techniques and methods. The subjects of such study usually involve well-qualified professionals with clear opinions and experience in their areas of interest [16].

Following this method, we have intended to collect and assess evidence of a sample of interest in the topic to deal with. Taking them as a basis, we will validate if the solutions provided coincide with the evaluations of the participants and, this way, they will corroborate our contribution to the MDWE paradigm.

We have decided to apply this experimentation technique due to the number of users participating in it and thus, another more open method will result in a long and expensive study. Thanks the possibility of doing it on the Internet, we have preferred this method rather than some more complex and durable [17] ones.

4 Challenges

4.1 Lack of control in development and deployment [S, T, E]

Creating a Web application is a complex process that involves not only coding/modeling, but also having meetings with stakeholders and debugging to fix production problems, for instance. In particular, when the application is deployed, aspects such as monitoring, logging and profiling become more important [3]. Therefore, the engineering teams that develop Web applications must have the ownership and responsibility for the deliverables. Consequently, they need to control the complete process (from development to deployment).

Despite the benefits of model-driven technologies, they also add an extra level of complexity as the derivation process is often seen as a “magic wand” that obtains an application from a set of models. Most MDWE tools hide and make this process “close source” creating a dependency between the MDWE tool and development teams. This dependency does not match time constraints and the fact of not having the source code available to everyone will make things even worst. Additionally, unless the MDWE tooling is open source and it has a big community, which hardly ever happens, it is likely that the commitment from the MDWE tooling team will require paying fees in the form of licenses, what increases the cost of development.

4.1.1 Guideline

MDWE tools need to provide the following aspects in order to deeply control the actual development:

- Ways to hook in the modeling and transformation phases, so that developers can add and improve the development process based on the application they are building. This could be done if the transformation algorithm works using the template method design pattern.
- Make the code open source, so that the development team will debug the tool and the generated applications easily in case that these problems arise during the development or maintenance process.

4.2 Too focused in navigation [T]

Original hypermedia-based Web applications that were developed 10 years ago are rather different from the current integration Web paradigm, where navigation is one tiny concern. Aspects such as Rich Internet Applications (RIA), integration with different systems, search capabilities, personalization and recommendation have become more complex and relevant points than navigation itself.

In the early stages, MDWE approaches were created to adapt the navigation paradigm from hypermedia to the Web. Many new concepts were introduced and although most of these enhancements have been reported in the literature [18], only a few of them have been actually implemented. Therefore, today the main model of industry leading WebRatio tool is the navigational model that describes the navigational paths that a user can follow. Opposite to what many researches have shown about the importance of Web application aspects (e.g. integration), navigation is still the most important one for MDWE approaches.

4.2.1 Guideline

MDWE approaches need to detach from being focused on navigation to support it as one concern of many present in the Web application lifecycle and architecture. Furthermore, there is a special requirement to provide a real tool support for the features that MDWE approaches claim to have and, in case that they are not supported, provide hooks to perform manual coding of these features.

4.3 Metamodel support and adaptation of its tools [T, S]

The increasing amount of technologies that are being developed every day in addition to customer's time constraints pose multiple challenges to Web application development. Although these tasks can be performed manually (in code-based environments) and may be time consuming, they can be easily performed by extending them into the so-called "hot spots" that code-based frameworks provide.

In MDWE approaches, this kind of extensions may be either carried out by instantiating some pre-existing metamodel classes or, if the functionality is not supported, by metamodel extensions. Extending a metamodel does not only require to add new classes and transformations, but also to adapt tools in a timely manner. Some approaches, such as NDT [19] or UWE [20], are extensions of the UML profile, thus, these approaches can be adapted more easily than others, such as IFML [21].

4.3.1 Guideline

Due to the existing technology and economic time constraints, metamodels and tools need to be adapted in a few days to cope with customers' demands. Currently, changing metamodels and tools can take a few weeks and, thus, they cannot be accepted by the customer's timeline, specially in the context of use of Agile methodologies.

4.4 Traceability and debugability [T]

One of the most important aspects of software development is the ability to introspect, change and monitor the "live" application under development to quickly fix the problems; these actions are considered as "debugging" an application. Code-based development in other languages such as Smalltalk, Java and .NET has these features from the beginning, making easy to iterate them along the development process.

In MDWE tools, where code is generated from models, the ability to debug is related to the derivation of traceability links between models and the generated code. Nowadays, only WebRatio partially supports this schema by allowing debugging the application under development [22]. Nonetheless, WebRatio needs further work to help trace back the problems while the application is running in production. Unfortunately, exceptions occur as no support is provided in this case. All these aspects make core engineers avoid adopting MDWE tools as they lack control over the system under development.

4.4.1 Guideline

As high-level languages (e.g. Java) provide ways to trace back problems to concepts of the language (e.g. classes and line numbers in exception stack traces), the MDWE approach should provide those features in order to detect the root causes of the problems. Additionally, further debugging support must be given to debug, evaluate and alter the application while it is running in a development environment.

4.5 Lack of tools [T]

Building a Web application requires a set of tools that eases the process of development, deployment and monitoring. For example, a typical JEE Web application can be developed using Maven [23] as a build system, Jenkins [24] for continuous integration and a variety of frameworks to actually build and implement the application (Spring, Hibernate or AngularJS, among others). A set of tools that automate and control the process from the moment the application is built to the instantiation of the servers and application deployment, as well as its initial monitoring is provided in order to perform the actual deployment. All these tools, despite coming from a lower level of abstraction (if we compare it with the MDWE approach), clearly help construct and deploy an application.

However, the actual development process of applications from high-level models has been specifically pointed out in the MDWE area. Although this is correct in terms of the MDWE philosophy, it increases the effort made in development and monitoring for many other topics discussed in this work (technological features, traceability, and debugability). Below, we provide concrete examples that illustrate this point:

- There are no tools to link stack trace exceptions back to the model elements. This aspect complicates the task of correcting errors originated in the application deployed.
- There are no tools to support the monitoring probes of the model elements and therefore, detecting performance issues is also a difficult activity. By using monitoring technologies like

New Relic [25], we can identify some of these points, but these would be classes derived from the MDWE tool that may correspond to multiple model elements.

4.5.1 Guideline

MDWE approaches need to support a handful set of tools that help in the complete application lifecycle, in order to provide valid approach for industry. Some of the aforementioned tools (e.g. Maven, Jenkins or New Relic) are fully extensible, so building some of them can be fairly simple by having good traceability links and extending them with the right information. Tools around Web applications are one of the most critical aspects to keep the application running 24x7.

4.6 Rigid architecture [T]

Creating a Web application of any size may require small to big changes in a standard 3-tier Web application. Some elements that need to be considered may involve integration with external services, processing queued information in an asynchronous manner, exposure of Representational State Transfer (REST) services for external users or internal mobile applications, among others issues. Hence, being able to adapt architecture to support any of these types of requirements is extremely important.

In current MDWE tools, architecture is not modeled at all and, as a consequence, they derive a simplistic 1-tier Web application [26] that can only handle a few sets of use cases and does not allow the development team to be able to adapt to future needs. A recent paper [27] has shown the necessity to model these aspects in some way, so that they can be considered through the derivation process. In its actual state, MDWE tools can lead to simple applications that may not scale properly, thus making it harder to be adopted in medium to big size companies.

4.6.1 Guideline

Adaptability to more complex requirements that may involve functional (e.g. processing offline data) or non-functional (e.g. performance and scalability issues) requirements entails that the MDWE approach needs to model architecture in such a way that development teams can decide which approach should be used. We must stress that although some architectures (e.g. 3-tier Web app) can be pre-configured, it is crucial to model architecture primitives and let development teams abstract higher-level concepts from them, such as the 3-tier Web application, instead of hardcoding it.

4.7 Technological aspects [T]

Logging, caching, load balancing and profiling are some of the aspects that engineers need to build high-scalable Web applications [3]. The lack of any of them poses some limitations on the type of application that can be produced. For instance, the absence of a caching strategy forces the application to compute or fetch information for every request, thus limiting its growth. Additionally, it may add the following problems:

- Run out of DB connections: Not being able to cache information stored in a DB requires the usage of a DB connection for every request. Thus, because of a DB limitation issue, the maximum number of users able to access the Web application is equal to the number of DB connections and as a result, new users will not be able to access the DB.

- Increasing response time: If we cannot store external service calls, they need to happen every time, thus increasing the overall response time of the requested Web page.
- Increase in hardware needed: If no cache is provided, we may need to use more hardware to re-compute values that were computed before.

Even though caching is a fairly simple aspect intrinsic to application development, MDWE approaches consider it together with the aforementioned aspects as a "technological" element. Being part of this category means that little importance has being paid to model, thus engineers will have to tweak them in the generated code. As none of the MDWE tools provide a roundtrip between the generated code and models, these "technological" tweaks have to be adjusted every time the application is derived.

4.7.1 Guideline

"Technological" aspects need to be considered in some way inside the model-driven development. If the MDWE approach intends to consider and model them, it will provide a great benefit for the size and quality of the application that can be built with MDWE tools. At the same time, the more response time is reduced, bigger amount of work is handled by the same amount of hardware, what will clearly show the benefits of using a model-based solution.

4.8 Community [S, E]

In the MDWE research area, there is a good initiative like the MDWEnet [28], whose main research focus are meta-modeling and model transformations. It was created with the aims of improving interoperability between existing MDWE approaches and their tools, so as to offer better methods and solutions to industry. Today, most approaches have lots of aspects in which they differ[29]: differences between meta-models and models, unlike ways to implement transformations or diverse tools and used technology, among other things. Then, there is a lack of consensus and documentation among approach designers and this entire context is causing different situations:

- On the one hand, organizations do not know how they can take advantage of these approaches and how they can be optimized in their particular context, due to the diversity set of characteristics they offer and the global heterogeneity associated with specific aspects or ideas processed by each approach.
- On the other hand, under this situation it is very complicated for approach designers to identify the real requirements and demands of the organization in order to improve their approaches or design new ones.

Nevertheless, within the context of the MDWE approach, WebRatio is an exception. This tool support provides a big community with a big variety of tutorials, webinars, user guides, assistance, fora and different types of tool certifications for users. Nowadays, there is no doubt that WebRatio is the leader tool in the market within the context of MDWE approaches, although neither this tool nor this community can be compared to other communities in the world related to Web development.

Unlike MDWE approach, there are existing and very extended Web development frameworks around the world like Ruby on Rails [30], Django [31], Grails [32] or Codeigniter [33], among others. All of them involve big communities of developers that provide lots of documentation, tutorials, user guides, fora and assistance, for instance. Moreover, these frameworks share many features and components that let developers compare them.

4.8.1 Guideline

These limitations and problems of description regarding the MDWE approach not only entail understanding these issues, but also unifying criteria and defining common strategies in a shared quality model [29,34,35]. Besides, this common model could help approach designers when improving or designing new approaches in the future. Besides, a common model can enable developers to compare all these MDWE approaches.

4.9 Licensing [E]

Today, two main business models to exploit these approaches in industry have been found on the MDWE approach.

The first one consists in implementing a specific tool support for the approach from free source environments. This specific tool can be offered to organizations either by means of license fees or freely. With regard to the former, there are different types of charges for each tool depending on the particular case of the organization. According to existing license fees on the MDWE approach, we have classified costs into "High" (more than $5,000 by activated seat), "Medium" (between $5,000 and $1,000 by activated seat) and "Low" (less than $1,000 by activated seat). For instance, WebRatio is developed under a free source environment like Eclipse, but with extensions that transform the Eclipse environment in a practical and valuable tool to support IFML visual modeling standard. In this case, WebRatio offers an Enterprise Edition license and organizations must pay a fee for each activated seat. This charge can be classified as "High" cost.

The second one deals with implementing the approach under the context of a powerful but paid CASE tool support. Then, the use of the approach is free, but organizations must pay for its license. For example, the NDT-Suite is a supporting tool that is developed under the Enterprise Architect (EA) [36] environment. This CASE tool allows organizations to have and work with all elements of the NDT approach and, under the environment of EA, it enables them to work with concepts of the approach together with lots of other visual modeling diagrams and characteristics that it additionally offers. One activated seat of EA license can be classified as "Low" cost. Other example of this business model is MagicUWE, a tool that has been developed for the computer-aided design of Web applications using the UML-based Web Engineering approach. MagicUWE has been designed as a plugin of MagicDraw [37], which is a tool support whose cost can be classified as "Medium".

We must consider that organizations pay a fee for these licenses, only if they are guaranteed to receive the value they need with minimal costs, risks and incertitude. As regards tool support value, it is not possible to identify the most valuable approach because it depends on the context. This way, each tool support has its advantages and disadvantages. It is the context that lets us decide which approach is the most suitable one. These tools can be incorporated in industry by maximizing their competitiveness: $Competitiveness = Value\ /\ (Cost + Risk + Incertitude)$

In fact, there exist other recognized and used frameworks in the market for the development of Web applications that they are not so abstract approaches, but effective solutions for developers. Frameworks like Ruby on Rails or Django, which encourage rapid development and clean pragmatic designs for developers, are completely free. They also have broad user base and community that do not require paying for any kind of support license. Finally, one of the most important points to highlight is that they are currently very extended solutions for developers, what enhances companies' confidence to consider their use.

4.9.1 Guideline

Key factors are not only costs of a license, but also value (although it depends on the context), risks and incertitude that organizations assume with the implementation of these approaches and tools. Hence, it is important to offer a high-value solution, free of charges and minimal risk and incertitude for developers.

4.10 Summary

It should be noticed that several key points of the MDWE approach can be analyzed with the aim of improving the presented paradigm and allowing it to be applied in today's industry. Hence, it can show the capacity of evolution that the MDWE approach has as well as its potential development, if changes are applied for an improvement in Web development at present.

Essential aspects, such as: control during the development and deployment of the application, traceability and errors debugging, or rigidity of the architecture and technical aspects may affect the development of a project. As far as these aspects are concerned, this paper evaluates the aforementioned points and checks their level of acceptance.

In contrast, there are other aspects that are not focused on the development or deployment of Web application, such as licenses, support community, navigation or supported metamodel, that must also be evaluated so as to implement other phases of the lifecycle.

5 Analysis of survey data

After establishing a series of hypotheses to the problems set out, it is necessary to carry out a study to gather information about the subject matter from experts' experience in the field in order to be able to compare their knowledge and evaluate results of their opinions. For this purpose, a survey has been carried out among a number of IT professionals from companies such as: Ayesa, Endesa, Everis, INPRO, Fujitsu, ATOS or Airbus D & S, among others. In particular, 50 users from those companies have participated in the project. The objective of this survey is:

- Firstly, to obtain information about the participants as well as the company for which they are working. This makes us differentiate among the different members surveyed.
- Secondly, to analyze the processes or procedures together with the tools they use in their companies and their opinions.

5.1 Sample analysis

To begin with, participants in the survey are studied in order to know more information about the company they work for. The first question concerns their experience, essential and relevant information for the final result of the survey. On the one hand, we can confirm that a large number of participants have many years of expertise in the sector. This allows us to obtain results endorsed by long-distance professionals who still hold positions very close to the questions asked in this survey, such as Programmer Analyst, Project Managers or Senior Tester. On the other hand, that experience also enables us to gather more knowledge of the company in which they are working, since they have enough years of expertise to know how the company runs.

In contrast, the other participants can also provide the approaches of the staff that sum up less time in the labor market or in the company. Thus, we can check the different approaches of the selected sample.

Another question posed has to do with their academic positions. In this case, there is more number of graduates among the most experienced people than in the less experience group. Therefore, we can emphasize that there is a high-level knowledge among experts selected for the survey. In addition, most of the participants do not have any certification oriented to the development of Web applications.

Moreover, we analyze the different companies from which the participants come. Approximately half of the organizations evaluated are companies that sell products based on Software, which allows us to obtain data from a sample with knowledge alienated with the issue of the problem that concerns us. Additionally, the remaining categorized subgroups have a number of participants needed to obtain information from other points of view that will further enrich the final data.

In order to better understand the sample to be evaluated, we assess the dimension of the evaluated companies. Most of these organizations have a very large number of staff. In light of this, we can consider that the results we get from the survey come from leading companies in the Software sector.

Finally, we also study the tools and technologies used by these organizations, which will serve to link these companies to the hypotheses presented here. On the one hand, participants are asked about the availability of a Web application development methodology or framework and most of the users answer that they do not have any standards for application development. On the other hand, methodologies such as SCRUM and Capability Maturity Model Integration (CMMI) are very much extended in organizations, as well.

As far as development languages are concerned, it is worth highlighting that the majority of companies use Java language. Furthermore, when they are asked about the use of a framework for automating the development of Web applications, only half of them use Spring, corresponding to Java language, as a framework of development.

To summarize, we have a sample with several subgroups that are well differentiated and specialized in this topic. We also find participants from companies with a volume large enough to support the results in their dimensions. These companies show two clearly differentiated aspects dealing with the use of frameworks for software development, so that they clearly operate in a different way in terms of development.

5.2 Analysis of results

To start with the survey, a series of questions have been raised in relation to the hypotheses set out above in order to obtain information about the topic that this article addresses. Results will be presented and a comparison with the hypotheses will be carried out.

First of all, a question arises about the architecture of the applications under development so as to know its degree of relevance in the projects.

As Figure 2 shows, most users consider that understanding the architecture of the application that is being developed is extremely important, specifically 69 of the users participating in the survey (54.8%) has marked this response. In general, 99.2% think that understanding the architecture of the application they are developing is essential, what shows that a consensus is reached on the majority of the sample surveyed from the opinion of most of the companies, which are evaluated through their participants.

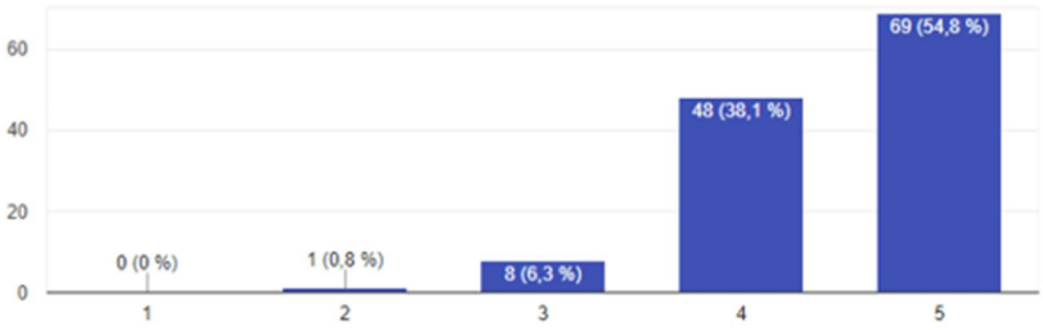

Figure 2: How important is for you to understand the architecture of the application you are developing?

The next question asked is to sort a series of activities according to the effort they entail in the development of their projects. Indicators range from activities that require greater effort, given value 1, to activities demanding less effort, given value 6.

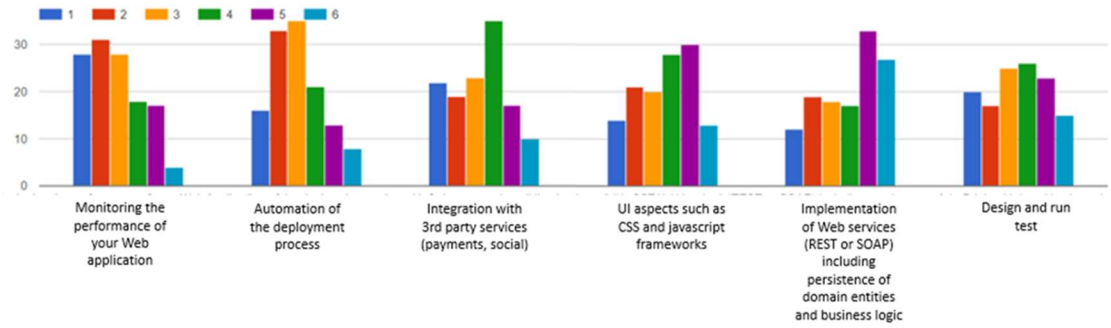

Figure 3: Please, rank the following activities according to the effort made in developing your projects.

First, we have analyzed the effort involved in designing the navigation of a Web application. Figure 3 shows that the results obtained are not totally helpful since the opinions of the users are quite different in this regard. The greater percentage is obtained in the middle zone of the chart, that is, an indicator of effort between 3 and 5.

Second, the next action to evaluate is to monitor the execution of the application. As in the previous chart, quite high results are obtained in the beginning, although more differentiated. In this case, indicators 1, 2 and 3 show that greater percentage of effort is obtained.

Third, the next activity to measure is the automation of the development process. The result shows similarities between indicators 2 and 3. It can similarly be observed that in Figure 3 other indicators are quite similar. Moreover, the chart related to the activity concerning integration of Web services is divided into three parts.

In this case, the most outstanding indicator is number 4, which represents a medium effort for most of the respondents. The following chart represents the development of aspects related to the user interface. It displays that the results with the highest percentage are those referring to indicators 4 and 5. Subsequently, the closest results are those corresponding to indicators 2 and 3.

Finally, the last chart displays the implementation of Web services (REST or Simple Object Access Protocol, SOAP) including the persistence of entity model and business logic. Again, the higher results are those corresponding to indicators 5 and 6 that cover most of the total. Once the results referring to the effort involved in the development activities of the projects are analyzed, we present the results regarding the time available to implement and put into production a requirement, when a change is requested during development.

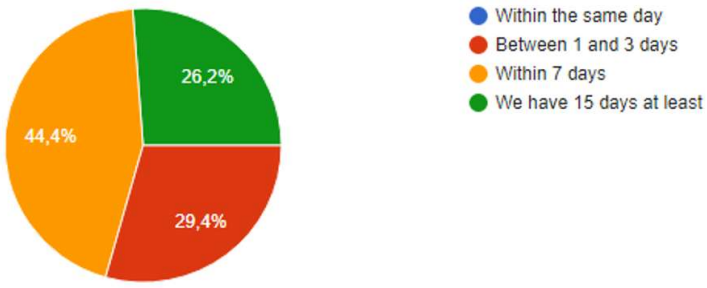

Figure 4: When a new feature or change request comes from the customer, how fast do you need to implement and deploy it to a production environment?

As Figure 4 shows, most of the users (44.4%) consider that the time available to implement and put into production a requirement when a change is requested during development is about 7 working days.

Below, it can be observed the results regarding cases where there are some incidences in production. For this purpose, we will evaluate the importance of having traceability of causality of the incidence.

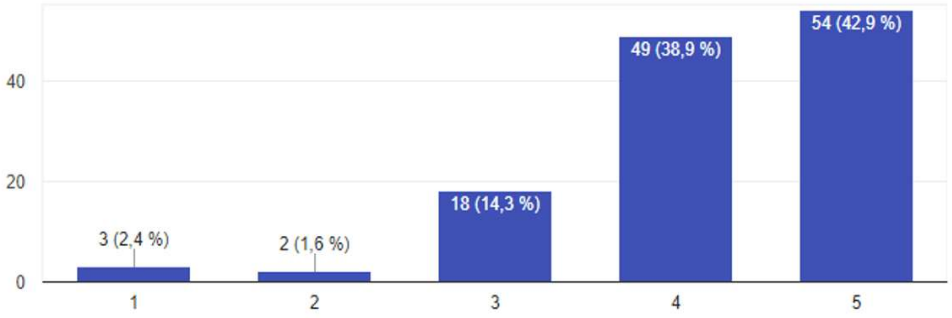

Figure 5: If you deal with a production problem, how important is for you to trace back the problem to the development environment?

Results in Figure 5 show that the majority of users consider that the fact of having traceability of the issues that could have caused an incidence is extremely important, being 42.9% the number of responses out of the total. That percentage together with 38.9% of the very important responses comprise 81.8% of the total responses, what remarks the weight given to this fact when dealing with incidents.

The next question posed to participants is to sort a series of elements according to the importance these assume in the development and maintenance of their projects. The indicators range from the most important elements, given value 1, to elements that have a minor relevance, given value 5.

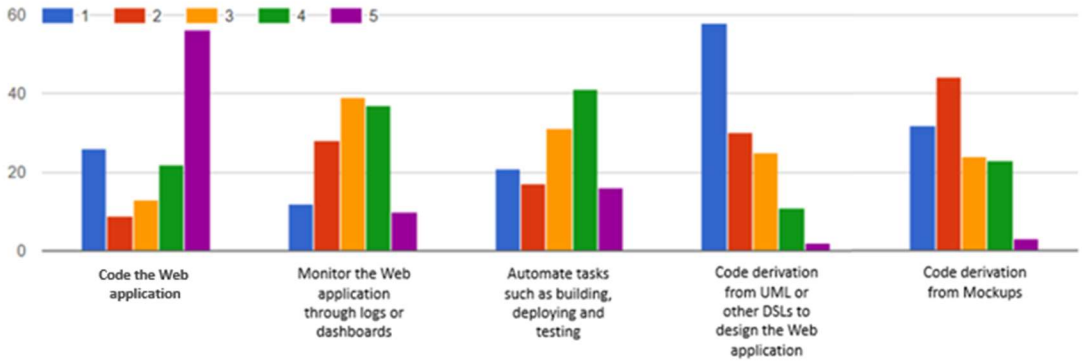

Figure 6: Rank the following items based on their importance for the development and maintenance of your Web applications

First, the importance of the code of Web application has been analyzed. Figure 6 indicates that the code of Web application is fundamental in a Web application.

Second, we present the results obtained in the query about monitoring Web application using logs or dashboards. In this chart, we can observe that the core values cover most of the users' responses, thus we can state that they have medium significance.

Third, we show the results obtained in the automation of tasks such as construction, deployment and testing. Regarding this subject, the participants think that these elements are important, but not the most important ones, since most of the answers are concentrated on the middle area of the chart.

Fourth, we present the generation of code from UML diagrams or other models used to design the application. Figure 6 represents that code generation is not among users' main priorities.

To conclude with the question about ordering a series of elements according to their level of importance in the development and maintenance of projects, we present the results obtained for the generation of code from mockups or screen prototypes. Now, all indicators except for 5 show similar results, out of which indicator 3 stands out.

Another question asked in the surveys is the importance of knowing and controlling the design of a Web application. Results are shown in Figure 7 below:

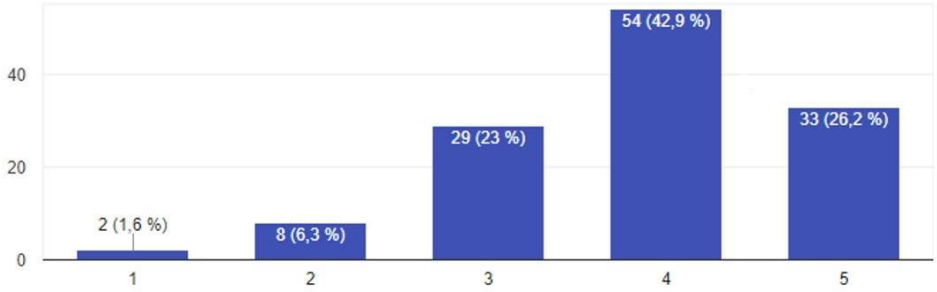

Figure 7: How important is for you to define the design of your Web application?

As we can observe in Figure 7, the set comprised in 92.1% of the sample considers that it is key to know and control absolutely Web application architecture, highlighting it as the most chosen and relevant option.

The next question to the participants is to order a series of phases and activities according to the effort they need to be addressed. The indicators range from activities or phases that require greater effort, given value 1, to activities or phases that demand less effort, given value 6.

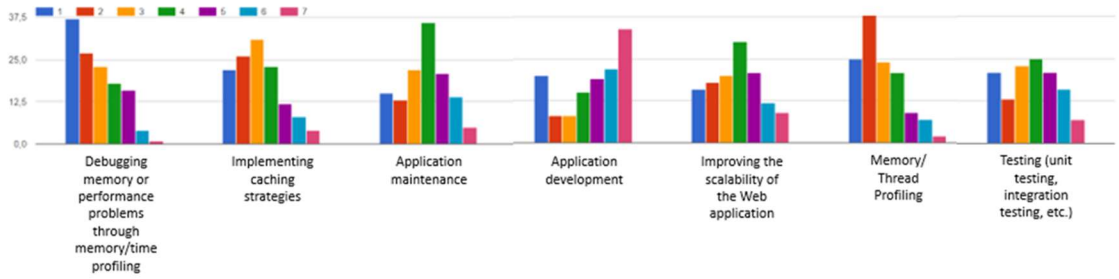

Figure 8: Please, rank the following aspects according to the effort made on addressing each development iteration

First, we have analyzed the effort made in debugging memory or performance problems. Figure 8 shows that we are facing an activity that needs a great effort for participants to be addressed, since the highest percentage of results is found in the first indicators.

Second, another activity to be evaluated is the implementation of caching strategies. The vast majority of the results are concentrated in the lowest indicators, what points out that this task is not too expensive for users.

Third, we present the effort that maintaining the application supposes to participants. Figure 8 confirms that the results are concentrated in the intermediate indicators, where indicator 4 points out that the effort of this task is not extremely expensive.

Fourth, we show now the effort made in the development of the application. The results of this chart are equitable, highlighting indicator 5 as the highest value, followed by indicator 1, as the second most selected value.

Fifth, another important aspect to look at is the improvement of scalability in the application. The results are clearly located in the middle of the chart, what is a symptom that for participants it is a task that involves medium-high effort.

Following the presentation of results obtained in the surveys, we describe the effort that the participants estimate the profiling task or analysis of the application performance involves. In Figure 8, the results are concentrated in the initial zone where indicator number 2 stands out.

Finally, we show the results obtained for one of the key phases in software development: testing. This chart presents equitable values where it is not possible to clarify a response among all the indicators obtained.

After analyzing the effort of the development phases in companies, it will be verified the importance that the participants give to the existence of communities and tutorials about tools and frameworks that are usually used in software development.

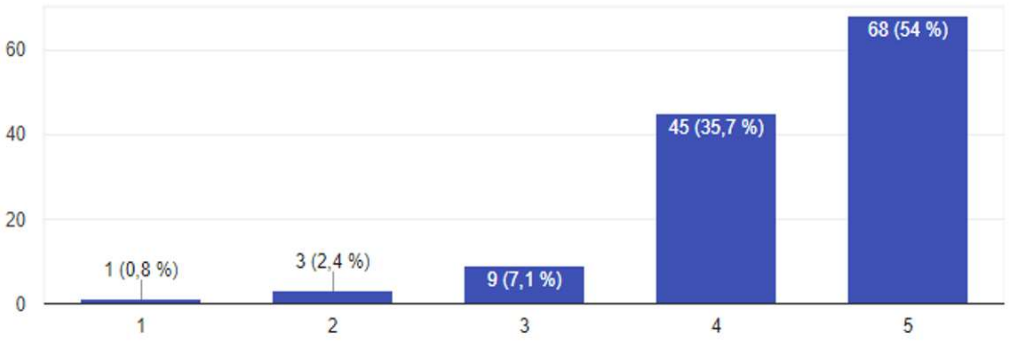

Figure 9: How important is the community and tutorials around the tools and frameworks you use for development?

Figure 9 represents that the values are concentrated in the highest options, with 89.7% of the options being selected for values "Very Important" and "Extremely Important".

To finish with the presentation of data, there is a possibility that concerns whether the organization would be willing to pay for a license of a framework or tool that may add some value to the development.

It is clearly observed that the percentages displayed in Figure 10 indicate that all companies prevent from investing too much in tools, especially in software, since they fall into disuse too early. For this reason, 73% of the participants condition the purchase of such license to the value that contributes the equally as well as its profitability.

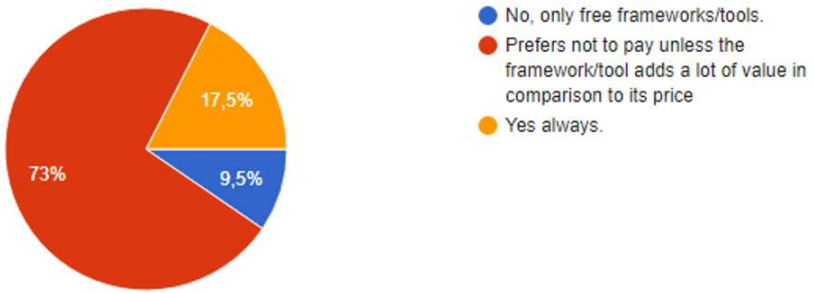

Figure 10: Is your organization/customer willing to pay a license to use a framework or tool that provides some value to development?

6 Validation of results

Once the results of the surveys have been assessed and presented, we will compare the conclusions obtained with the hypotheses presented in section 3 (Challenges).

6.1 Model extensibility through hooks

To begin with, we will compare conclusions from the survey with the hypothesis dealing with the use of design patterns by means of the template method, as well as the conversion of code into an open source. For this purpose, we will look specifically at the charts of Automation of the development process where it is confirmed that the effort made in this task is relatively significant. We will also take into account the results obtained in the charts of code generation from other elements such as mockups, where it is observed that the result of the survey gives this fact a considerable importance.

6.2 Open source

As for open source, it is critical to highlight the chart Communities and tutorials on tools and frameworks in development, where 80% of results state that they are "very important". To conclude this hypothesis, it should be added that organizations prefer not to pay for a license of a tool unless the value it offers will generate profitable outcomes, as indicated by 73% of the results in the last chart.

6.3 Need to include navigation

Continuing with the validation of the hypothesis, we proceed to verify the need to incorporate navigation in the Web lifecycle, as well as a tool that supports it.

In this case, it has not been possible to obtain relevant data indicating the need to include this aspect in the lifecycle.

6.4 Time of the client's demand

The next hypothesis that is going to be checked refers to the time of the client's demands and how they can influence tools such as metamodels. This can be noticed in the chart of Request for change of a

requirement during development, where the highest result (44.4%) is given to “7 business days” as the maximum period to respond to a request.

Then, error detection features are included in the MDWE approach by means of the following charts:

- Charts regarding Traceability of the elements that could cause an incidence that let us prove that the majority of participants (81.8%) consider it as a "very important" aspect.
- Monitoring of Web application through logs or dashboards that is of average importance among participants.

6.5 Tool to support the complete lifecycle

Next, we will check the need to find a tool capable of being modified in order to support the complete lifecycle.

6.6 Adaptive architecture

Later, the need to have an adaptive architecture, that allows the development team to abstract high-level concepts through graphs that contemplate the importance of understanding and controlling the architecture of an application, will be corroborated. This architecture motivates that most of the users set out the idea of understanding this concept as an important aspect.

6.7 Technical aspects that the MDWE approach supports

Subsequently, the technical aspects that the MDWE approach supports are verified. For that purpose, some charts corresponding to the measurement of the effort of the technical activities are used. They emphasize that all the charts center their highest values on their middle zone.

6.8 Need of communities and tutorials

One of the last validations will confirm that communities and tutorials really need tools and frameworks as indicated in the chart on this topic, where most respondents agree that it is a crucial and essential aspect.

6.9 Licenses

Finally, the topic of validations is completed with the hypothesis on the subject of licenses and the value they provide. The last chart expresses truthfully, as 73% of participants chose this option, that organizations prefer not to pay, unless the value offered by such tools is profitable enough.

Finally, Table 2 presents a summary of this survey with the results of each of the proposals.

Challenges	**Results**
Model extensibility through hooks	The effort made in this task is relatively big and important. Code generation is highly relevant
Open source	Organizations prefer not to pay for a license of a tool
Need to include navigation	It has not been possible to obtain relevant data
Time of the client’s demands	Request for changing a requirement during development. 7 labor days as the maximum term to respond to that request. With regard to validation of the inclusion of the error detection, the most important aspects concern traceability and Web monitoring with logs

Tool to support the complete lifecycle	A tool capable of being modified in order to support the complete lifecycle
Adaptive architecture	The majority of users consider that it is very important understanding this concept
Technical aspects that the MDWE approach supports	The highest values are intermediate-value effort
Need of communities and tutorials	Necessary and important
Licenses	Prefer not to pay for them

Table 2. Summary of Survey

6.10 Threats to validity

There are several threats to validity that are considered during the experiment design. This research presents a preliminary result and, for space limitations, we only include the most important set of identified threats.

Construct validity. The experiment was designed to test a set of hypothesis based on a well-defined survey. In order to reduce the experiment's complexity and bias introduction possibility, we defined the survey as the only variable. Questions were non-ambiguous and a classic 5-point scale answer with no free content was specified to mitigate any misunderstanding.

Internal validity. To mitigate these kinds of threats, a homogeneous set of professionals were chosen and filtered in order to avoid any external factor that may influence the answers. We pre-checked that all participants had several years of expertise in the sector and also ensured that they had valuable industrial positions at that moment. Besides, several different companies were chosen in order to respond to diversity coming from diverse workplaces with internal practices that may vary from standards.

External validity. MDWE methodologies are 15 years old and, as model-driven paradigm in general, they have demonstrated that there exist cases in which models can increase productivity significantly. However, MDWE methods have not reached as big audiences as others, for instance, Agile. While this may compromise generalization of the results, this paper intends to answer a bigger question that is general in the Computer Science field: how much and in which contexts abstraction helps increase productivity in the different phases of the development cycle? This paper has the intention to answer this question, which is a general one, in the context of Web Engineering. To lessen this kind of issues and ensure generalization of this survey, our questions were carefully elaborated to include basic and core aspects of the Web development process like architecture, logging, maintenance or profiling, that are extremely important in enterprise contexts.

Conclusion validity. Every hypothesis described in Section 3 was backed up by concrete results of the survey, whose questions were carefully designed to prove them in a Goal-Question-Metric-like fashion. Statistical significance was not a problem in this case since no statistical tests was required for this work.

7 Conclusion and Future Works

To conclude, it has been possible to successfully verify many of the proposals raised in section 2 thanks to the participants of the surveys that share the presented needs. In spite of this, not all the proposals have had the expected acceptance, since the needs of the respondents have not really clarified whether the proposed solution fits each of them or it is the same for all. Nevertheless, after carrying out the surveys and due to the possibility of improving some of the phases of the MDWE approaches with the

proposals presented in this article, it can be verified that the acceptance and margin for improvement of this paradigm can be quite broad.

After the results obtained from the hypotheses posed, a new path can be defined in the future of the MDWE approach as a future line of work. In this sense, the idea would be either to implement such solutions so that they can be integrated into the MDWE methodology by means of some tool or to expand the existing ones. Apart from this, it should be noticed that the solutions proposed have not matched the results of the surveys, thus it would be useful to provide new solutions to overcome identified deficiencies or even to evaluate again whether or not such deficiencies really exist.

As we have realized throughout the paper, there are many fields that can be implemented within this approach, although we have just focused on some of them. As possible future work, it would be very interesting to cover other fields such as security, usability or support for testing among others, which would further increase the benefits of this approach.

Acknowledgements

This research has been supported by POLOLAS project (TIN2016-76956-C3-2-R) of the Spanish Ministry of Economy and Competitiveness and by the Fifth Internal Research Plan (VPPI) of the University of Seville.

References

[1] B. Selic, The pragmatics of model-driven development, IEEE Softw. 20 (2003) 19–25. doi:10.1109/MS.2003.1231146.

[2] G. Rossi, O. Pastor, D. Schwabe, L. Olsina, Web engineering: modelling and implementing web applications, Springer Science & Business Media, 2007.

[3] S. Hull, 20 obstacles to scalability, Commun. ACM. 56 (2013) 54. doi:10.1145/2500468.2500475.

[4] F.J. Domínguez-Mayo, M.J. Escalona, M. Mejías, Quality issues on model-driven web engineering methodologies, in: Inf. Syst. Dev. Asian Exp., 2011: pp. 295–306. doi:10.1007/978-1-4419-7355-9_25.

[5] M. Urbieta, D. Distante, J.M. Rivero, S. Firmenich, 25 Years of Model-Driven Web Engineering: What we achieved, What is missing, CLEI Electron. J. 19 (2016) 1:1-1:29. doi:10.19153/cleiej.19.3.1.

[6] S. Casteleyn, W. Van Woensel, K. Van Der Sluijs, G.J. Houben, Aspect-oriented adaptation specification in web information systems: A semantics-based approach, New Rev. Hypermedia Multimed. 15 (2009) 39–71. doi:10.1080/13614560902818297.

[7] I. Jacobson, S. Bylund, The road to the unified software development process, Cambridge University Press, 2000.

[8] N. Moreno, A. Vallecillo, Towards interoperable Web engineering methods, J. Am. Soc. Inf. Sci. Technol. 59 (2008) 1073–1092. doi:10.1002/asi.20811.

[9] J. Whittle, J. Hutchinson, M. Rouncefield, The state of practice in model-driven engineering, IEEE Softw. 31 (2014) 79–85. doi:10.1109/MS.2013.65.

[10] F.J. Domínguez-Mayo, M.J. Escalona, M. Mejías, M. Ross, G. Staples, Towards a homogeneous characterization of the model-driven web development methodologies, (2014). http://www.scopus.com/inward/record.url?eid=2-s2.0-84899720764&partnerID=MN8TOARS.

[11] J. Whittle, J. Hutchinson, M. Rouncefield, H. Burden, R. Heldal, Industrial adoption of model-

driven engineering: Are the tools really the problem?, in: Lect. Notes Comput. Sci. (Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), 2013: pp. 1–17. doi:10.1007/978-3-642-41533-3_1.

[12] J.A. García-García, M.J. Escalona, F.J. Domínguez-Mayo, A. Salido, NDT-Suite: A Methodological Tool Solution in the Model-Driven Engineering Paradigm, J. Softw. Eng. Appl. 7 (2014) 206–217. doi:10.4236/jsea.2014.74022.

[13] F.J. Domínguez-Mayo, M.J. Escalona, M. Mejías, QuEF (Quality Evaluation Framework) for model-driven web methodologies, in: Lect. Notes Comput. Sci. (Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), 2010: pp. 571–575. doi:10.1007/978-3-642-16985-4_57.

[14] R. Acerbis, A. Bongio, M. Brambilla, S. Butti, WebRatio 5: An Eclipse-based CASE tool for engineering Web applications, Web Eng. (2007) 501–505. doi:10.1007/978-3-540-73597-7_44.

[15] M.J. Escalona, G. Lopez, S. Vegas, L. Garccia-Borgoñon, J.A. Garcia-Garcia, N. Juristo, A Software Engineering Experiments to value MDE in testing. Learning Lessons, (n.d.).

[16] J.S. Molleri, K. Petersen, E. Mendes, Survey Guidelines in Software Engineering, in: Proc. 10th ACM/IEEE Int. Symp. Empir. Softw. Eng. Meas. - ESEM '16, 2016: pp. 1–6. doi:10.1145/2961111.2962619.

[17] K.B. Wright, Researching Internet-Based Populations: Advantages and Disadvantages of Online Survey Research, Online Questionnaire Authoring Software Packages, and Web Survey Services, J. Comput. Commun. 10 (2006) 00–00. doi:10.1111/j.1083-6101.2005.tb00259.x.

[18] E. Luna Robles, G. Rossi, I. Garrigós, WebSpec: A visual language for specifying interaction and navigation requirements in web applications, Requir. Eng. 16 (2011) 297–321. doi:10.1007/s00766-011-0124-1.

[19] M.J. Escalona, G. Aragón, NDT. A model-driven approach for web requirements, IEEE Trans. Softw. Eng. 34 (2008) 377–394. doi:10.1109/TSE.2008.27.

[20] M. Busch, N. Koch, MagicUWE - A case tool plugin for modeling web applications, in: Lect. Notes Comput. Sci. (Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), 2009: pp. 505–508. doi:10.1007/978-3-642-02818-2_49.

[21] M. Brambilla, IFML: Building the front-end of web and mobile applications with omg's interaction flow modeling language, 2014.

[22] P. Fraternali, M. Tisi, Using traceability links and higher-order transformations for easing regression testing of web applications, J. Web Eng. 10 (2011) 1–20.

[23] Maven, http://maven.apache.org, in: Last Access Febr., 2018.

[24] Jenkins, http://jenkins-ci.org, in: Last Access Febr., 2018.

[25] NewRelic, http://newrelic.com, in: Last Access Febr., 2018.

[26] A.O. Ramirez, Three-Tier Architecture, Linux J. 2000 (2000) 1–4.

[27] G. Toffetti, Web engineering for Cloud computing, in: Curr. Trends Web Eng., 2012: pp. 5–19. doi:10.1007/978-3-642-35623-0_2.

[28] MDWEnt, http://www.iswe-ev.de/activities/2007/mdwe/, in: Last Access Febr., 2018.

[29] F.J. Domínguez-Mayo, M.J. Escalona, M. Mejías, M. Ross, G. Staples, Quality evaluation for Model-Driven Web Engineering methodologies, Inf. Softw. Technol. 54 (2012) 1265–1282. doi:10.1016/j.infsof.2012.06.007.

[30] RubyOnRails, http://rubyonrails.org, in: Last Access Febr., 2018.

[31] DJango, http://djangoproject.com, in: Last Access Febr., 2018.

[32] Grails, http://grails.org, in: Last Access Febr., 2018.

[33] Codeigniter, http://ellislab.com/codeigniter, in: Last Access Febr., 2018.

[34] F.J. Dominguez-Mayo, M.J. Escalona, M. Mejias, a. H. Torres, A Quality Model in a Quality Evaluation Framework for MDWE methodologies, Res. Challenges Inf. Sci. (RCIS), 2010

Fourth Int. Conf. (2010). doi:10.1109/RCIS.2010.5507323.
[35] J.A. García-García, J. Victorio, L. García-Borgoñón, M.A. Barcelona, F.J. Domínguez-Mayo, M.J. Escalona, A Formal Demonstration of NDT-Quality: A Tool for Measuring the Quality using NDT Methodology, in: 21st Annu. Softw. Qual. Manag. Conf., 2013.
[36] EnterpriseArchitect, http://www.sparxsystems.com, in: Last Access Febr., 2018.
[37] MagicDraw, http://nomagic.com/products/magicdraw.html, in: Last Access Febr., 2018.