TOWARDS A HOMOGENEOUS CHARACTERIZATION OF THE MODEL-DRIVEN WEB DEVELOPMENT METHODOLOGIES

F.J. DOMÍNGUEZ-MAYO, M.J. ESCALONA, M. MEJÍAS

University of Seville, Seville, Spain
{fjdominguez, mjescalona, risoto}@us.es

M. ROSS

Southampton Solent University, Southampton, United Kingdom
margaret.ross@solent.ac.uk

G. STAPLES

British Computer Society (BCS) Software Quality Specialist Group, United Kingdom
Geoff.Staples@bcs.org.uk

In recent years a large number of Model-Driven Web development approaches have been designed and are being applied with success in real environments. However, as new ones are frequently emerging in this changing time, authors have to change and update them constantly and, consequently; development teams do not know which is the most suitable for them because in many cases it depends on their project scope. Furthermore, approaches are usually appearing with different concepts and terminologies in many cases, although all lack the use of standards and practical experience. Thus, the need of managing quality in this type of approach arises every day. This paper suggests a characterization of these methodologies in order to use this information for the quality management of Model-Driven Web development methodologies for authors and development teams alike. In addition, an experimental study in order to analyse and evaluate a Model-Driven Web development methodology (the NDT methodology) has been carried out within a specific work context.

Key words: Web engineering methodologies, Measurement, Software Quality

1 Introduction

Web development is currently an important task to take into account since Web applications are becoming more developed every day. In this context, The Model-Driven Engineering (MDE) paradigm plays a key role because it aims to increase the return a company derives from its software development effort basically by using models and automatic transformations. In this regard, the Object Management Group (OMG) has introduced Model Driven Architecture (MDA) which is an approach

for achieving the concept of platform independence and models may have the quality of being independent from any technological platform. MDE is a paradigm that will change the way an organization designs and develops software by separating an application's business logic from the infrastructure on which it runs. MDE principles are being used to successfully address the construction, evolution and adaptation of Web applications.

The growing interest in the Internet has led to the making of a large number of proposals [39; 23] which offer a frame of reference for the Web environment. MDWE (Model-Driven Web Engineering) is the application of the Model-Driven paradigm to the domain of Web software development, where it is particularly helpful due to the continuous evolution of Web technologies and platforms. Different concerns of Web applications are captured by using separate models i.e. for the content, navigation, process and presentation concerns. Whereas code comprises Web pages, configuration data for Web frameworks as well as traditional code, models are integrated and transformed into code [11].

During the last years, the Web engineering community has proposed several different methodologies for Modeling Web applications with different concepts and definitions such as UWE (UML-based Web Engineering) [23], WebML (The Web Modeling Language)[8], OOH4RIA[25], RUX-Method [33] or NDT (Navigational Development Techniques) [11] methodology among others. There is no standard consensus among them, but a lack in the use of standards and scarcity of both practical experience and tool support. In fact, every methodology has a set of advantages and disadvantages [39] that depends on the use-context or project scope.

Besides, given the large number of approaches available, it is not only necessary to evaluate the quality of the existing methodologies, but also to find out how it can be improved so that authors and development teams can be provided with helpful information. The first point to consider is the authors' view; they need to analyze, control, evaluate and improve MDWE approaches. The other point of view to take into account is development teams', who need to compare MDWE approaches (depending on project use-context or project scope) to decide on the most suitable one for them.

Surveys and comparative studies [27, 39, 12, 37] conclude that there are serious gaps. Today most approaches are not agreed in all their aspects, for instance: meta-models and models are different, transformations are implemented in different ways, some MDWE approaches cover all levels of abstractions (CIM, PIM, SPM, and code) and others only some of them, they use different tools and each MDWE approach is carried out in a different way.

Then, some of these approaches involve most of the levels of abstraction and they even have tools that support the automation of transformations in development processes. However, there is no control on what these approaches offer to development teams. In addition, in most cases it depends on the project scope and use-context and, in the face of this situation, an important need to assess the quality of existing MDWE approaches arises. Development teams that have to improve web systems do not know how they can take advantage of these approaches and how they can be helped in their particular context. The diversity features within the design of these approaches confirm the global heterogeneity associated with specific aspects or ideas processed by each approach. These limitations and problem of description not only entail understanding the proposed value, but also require an objective criterion for the improvement or the possibility of unifying criteria when designing new

approaches in the future. It is known that "you can't improve what you can't measure", which means that context has to be described in order for it to be measured and controlled. Thus, by measuring the environment, you can control and better it because you know what it needs to be improved.

This paper is organized as follows: In Section 2, a global analysis of the situation together with the related works and discussions about the issue is presented. Section 3 presents concepts such as the elements for describing the MDWE methodologies under consideration. In Section 4, the way of analyzing the methodologies in terms of their properties and using checklists for this purpose is described. In Section 5, in order to illustrate this evaluation process, a Model-Driven Web development methodology is evaluated to discover the state of completeness of the methodology. Finally, a set of conclusions, contributions and possible future work are stated in Section 6.

2 Related Works and Discussion

As far as standard methods for the measurement process are concerned, the ISO/IEC 15939:2007 [17] defines a measurement process applicable to software engineering and management disciplines. Firstly, the process is described through a model which defines the activities of the measurement process that are required to adequately specify what measurement information is required. Secondly, it shows how the measures and analysis results have to be applied, and finally, it examines how to determine if the analysis results are valid. The measurement process is flexible, tailorable, and adaptable to the needs of different development teams. ISO/IEC 15939:2007 identifies a process which supports the definition of a suitable set of measures that addresses specific Properties. It determines the activities and tasks that are necessary to successfully identify, define, select, apply and improve measurement within an overall project or organizational measurement structure. It also provides definitions for measurement terms commonly used within the system and software industries.

ISO/IEC 25040:2011 [18] contains requirements and recommendations for the evaluation of software product quality and clarifies the general concepts. It provides a process description for evaluating software product quality and states the requirements for the application of this process. The evaluation process can be used for different purposes and approaches. The process can be used for the evaluation of the quality of pre-developed software, commercial-off-the-shelf software or custom software and can be used during or after the development process. They describe neither methods for evaluating software production processes nor methods for cost prediction (software product quality measurements may, of course, be used for both of these purposes). However, these standard models are difficult to implement. The main weakness of standard models lies in that they are unable to explain how they are to be implemented. That is, it does not detail what Property you have to use or what is the most appropriate one, nor how to group value Properties for higher-level Properties or what the thresholds for each Property are.

In order to define metrics, GQM (Goal Question Metric) [3,4] is an approach to guide the definition of metrics. It is typically described in the open literature in terms of a six-step process, where the first three steps deal with using business goals to drive the identification of the right metrics and the last three with gathering the measurement data and making effective use of the measurement results to drive decision making and improvements. As far as measurement and evaluation is concerned, García et al. [14] proposes an approach to enable the management of software process measurement. The evaluation of software processes involves the measurement of a great diversity of

entities, from the models of the process of projects to resources and the products obtained. The proposal allows the integrated management of the measurement of these kinds of entities.

There are also several proposals for the metrics and indicators domain in the literature, one of them is [1] where a set of design metrics proposed for assessing the size and structural complexity of navigational models is discussed. Becker and Olsina [5] propose the INCAMI framework, which is an organizational purpose-oriented measurement and evaluation framework that enables consistently saving not only meta-data of metrics and indicators but also values (data sets) for concrete real-world measurement and evaluation projects. The INCAMI framework is made up of five main conceptual components namely, the requirement, measurement, and evaluation of projects definition; the nonfunctional requirements definition and specification; the measurement design and execution; the evaluation design and execution and the conclusions and recommendation.

In the MDE (Model-Driven Engineering) domain, some studies of Mohagheghi [26] discussed the Characteristics of MDE that are important when building quality, and it stated that the quality of models is affected by the quality of different Features such as modeling languages, tools, Modeling processes, the knowledge and experience of development teams/authors and the quality assurance techniques applied. In this sense, a set of Web methodology Features has to be described in order to carry out a quality evaluation of these types of methodologies.

Nowadays, producing faster and cheaper software of higher quality is critical in the software industry and the use of a MDWE methodology and its influence on the final product quality is an issue that must be acknowledged. The use of a methodology based on MDE is essential to achieve this aim. Up to now, the main criticism of the quality assessment process has little evidence drawn from stakeholders as to whether MDE meets the process reference model to ensure product quality.

Maibaum and Wassying [24] commented that quality assessments should be based on circumstantial evidence inferred from the process. In fact, the standardization of processes ensures consistency in their output, which may even institutionalize the creation of bad products. In summary, a standard process does not necessarily conclude with a quality product [32]. Therefore, it is not only essential to describe the methods or processes of methodologies, but also to describe the product and define the relationships among all methods and each product's properties in the results. In this paper, Web methodologies are described, considering the process as activities, artifacts used during the activities and techniques. So in future papers, the product's properties have to be described and related to the methods of methodologies.

In order to evaluate quality, according to Cachero et al. [6], it is necessary to count on instruments that are based on clear definitions. One of these instruments is a quality model. It is defined in ISO as the set of characteristics and relationships among them, which provides the basis for specifying quality requirements and evaluating quality. There are some models in the literature such as the WQM model (Web Quality Model) [7]. This model is introduced and distinguishes three dimensions related to Web features, lifecycle processes and quality features including the most relevant Web metrics using the framework, which is classified.

Therefore, to carry out quality management, it is necessary to define a quality model that identifies the set of characteristics and relationships among them. .Giving a definition of all these elements is not an easy task and some authors do not have a strategy to identify quality management

targets. Besides, a problem may arise if the quality model is not clearly designed or defined, since there is no goal to achieve.

In previous papers, a set of quality characteristics and an evaluation process for them is proposed [10]. Further, a framework (QuEF) to analyze, evaluate and improve quality of MDWE approaches is proposed [9] to cover a complete life cycle for the quality model. Consequently, to define a quality model to make these approaches in real contexts more effective and efficient becomes essential, as well as to achieve suitable tools to analyze, evaluate and plan the improvement of MDWE approaches automatically based on the quality model lifecycle management. In addition, it is important for authors to weight all these elements (properties, quality characteristics and influences among them) in order to give development teams what actually matters to them.

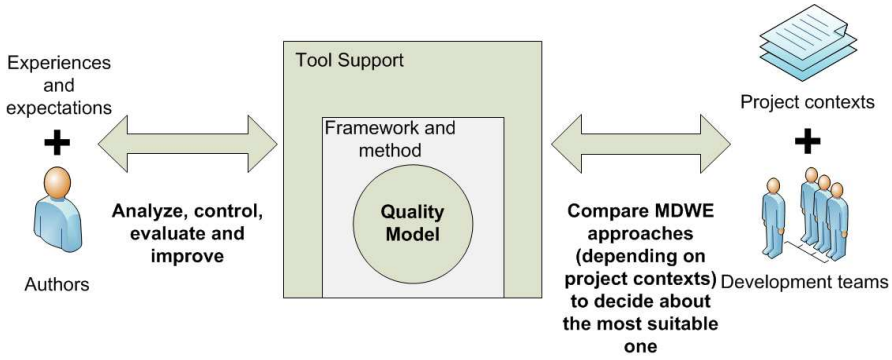

Figure 1 Conceptual scheme representing the goals to be achieved with QuEF for Model-Driven Web development methodologies

QuEF can be used by authors to analyze, evaluate, control and increase the quality of MDWE approaches and improve their design and results. In addition, this framework can be used by development teams to identify the most suitable one for them and decide which one will be used.

As shown in figure 1, the framework will be used from two points of view: authors', who need to analyze, control, evaluate and improve MDWE approaches and development teams', who need to compare MDWE approaches (depending on project contexts) to decide on the most suitable one for them. These objectives are detailed in the following sections. With QuEF we can get quality management in a systematic way, automating the quality management for this type of approach and any entity in order to reduce costs and time and improve quality within the quality management process.

The main difference from other frameworks is that the quality management is focused on the quality model life cycle as shown in figure 2. This means that it comprises several phases which include different objectives and artifacts. The above mentioned phases are the Quality Model Strategy phase, the Quality Model Design phase, the Quality Model Operation phase, the Quality Model Transition phase and the Quality Continual Improvement phase. Each one of them has a specific objective:

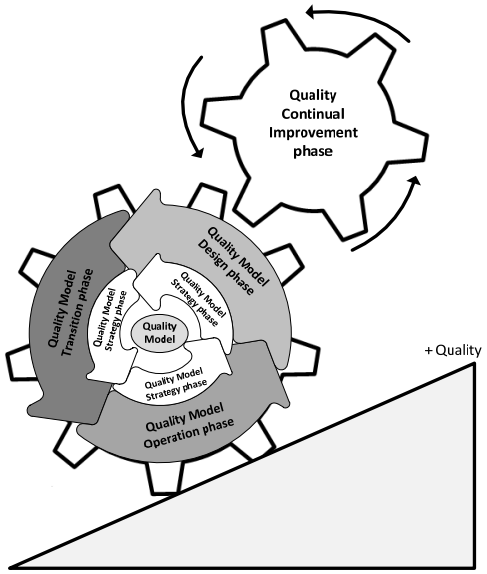

Figure 2 Quality Management based on the quality model life cycle [9]

- Quality Model Strategy phase: This phase is a strategic active that focuses on the definition of a strategy for the quality management. The past, the present and future view elements of the quality model in the domain under study are fundamental to achieve effective and efficient quality management.
- Quality Model Design phase: This phase is where the quality model is finally designed in terms of all strategic actives in the previous phase. This quality model is the model used in the next phase for operating for the quality management.
- Quality Model Operation phase: In this phase the quality model is used to carry out the Quality management. The Analysis and Evaluation management processes are performed within this phase.
- Quality Model Transition phase: If the domain or context is changed because of the appearance of new trends, then this phase describes the processes that carry out the changes in the quality model but without affecting the Operation phase.
- Quality Continual Improvement phase: This phase performs all processes to improve quality of all processes in the life cycle and also the quality model.

The main difference is that the framework defines a life cycle for the quality model, which is the focus of the quality management and all phases revolve around it. For an effective and efficient quality management it is essential to establish the domain under study. This is important not only for the quality management of these kinds of methodologies but to be applied to any

context or domain. For instance, even in developing a Web Application (a product) it is important to consider what type of product is concerned. It is not the same to develop an application or product for a bank to whom security stands as a quality characteristic imperative, as to develop a search engine which the usability or performance is crucial. So, a general approach for MDE is not enough for solving the problem of the quality management for MDWE methodologies. The purpose of QuEF is not only to assure a clear strategy for the quality management but, in addition, continuous automatic quality improvement by means of generating checklists and documentation, as well as automatic evaluations and plans in order to control and improve quality and thus, automatically, reduce effort and time.

All elements of the quality model are explained, although in this paper we focus on properties of MDWE methodologies (classified by features and sub-features). Once this quality model metamodel is explained, we pass onto Section 4, where some formulas are defined for the analysis and the evaluation processes, and which shows the results of the characterization of these methodologies. Section 5 illustrates a evaluation process for a Model-Driven Web development methodology which is evaluated to find the state of completeness of the methodology and improve its weaknesses.

3 Concepts in use

In ISO 15939 [17], the measurement information model constitutes a structure linking information needs to the relevant entities and attributes of concern. Entities include processes, products, projects and resources. The measurement information model describes how the relevant attributes are quantified and transformed into indicators that provide the basis for decision-making. The selection or definition of appropriate measures to address an information need begins with a measurable concept: an idea of which measurable attributes are related to an information need and how they are related. The measurement planner defines measurement constructs that associate these attributes with a specified information need. This measurement information model identifies basic terms and concepts. The measurement information model helps determine the measurement planner needs to specify during measurement planning, performance and evaluation.

The entity in our quality model represents a MDWE approach that has to be characterized by measuring its attributes. An entity may have one or more interesting properties to meet the information needs. In practice, an entity can be classified into more than one of the above categories.

In this paper, a quality model consists of a set of elements and the relationship among them, which lays the foundations for quality management. The quality model may be defined as “conformance to requirements” and/or “fitness of use”. This quality model contains:

- Properties, that is, the descriptive environment in which the quality management is going to be performed and the needs offered by authors to development teams of MDWE approaches.
- Quality characteristics are those quality aspects that authors of MDWE approaches must ensure in the set of properties offered to development teams[10]

In simple terms, all authors must be well aware of properties (they are the description of approaches and development teams’ needs and expectations to be covered), quality characteristics to be assured and the impact on quality characteristics, strategic quality management and the contribution of this strategy towards achieving the goal.

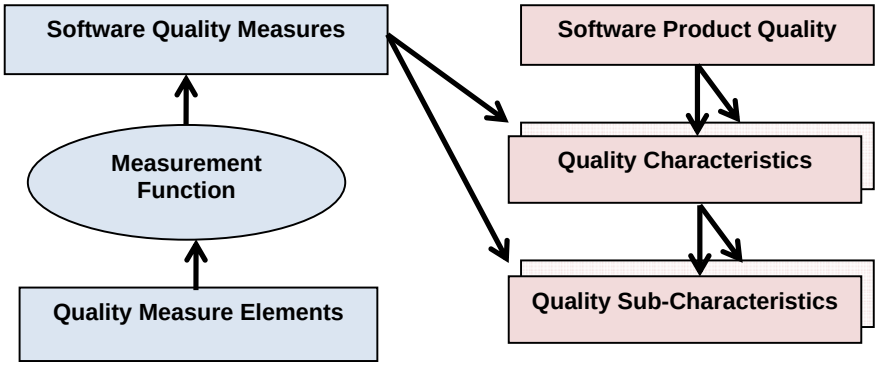

Figure 3 Quality Measure Elements Concept in the Software Product Quality Measurement Reference Model [17]

Figure 3 shows the relationship [2] between the quality measure elements and the software quality measures, and between the software quality measures and the quality characteristics and subcharacteristics. In metrology, these would correspond to base measures and derived measures, respectively. It can be observed that these measures, particularly derived measures, are specifically defined to measure the sub-characteristics of internal and external quality or the characteristics of quality in use. None of these is directly related to the software quality top level (which is itself broken down into three models, then into sixteen characteristics and finally, into a large number of sub-characteristics)

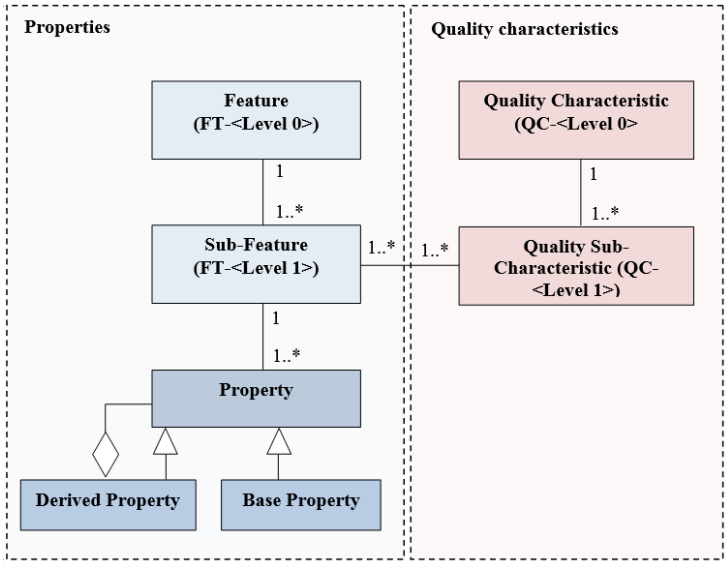

Figure 4 quality model metamodel for MDWE approaches

The following definitions come from ISO standard on software measurement process, ISO 15939 [17], which is itself based on the definitions in ISO International Vocabulary of Basic and General Terms in Metrology [19]. Further, in ISO 15939 [17], the type of measurement method depends on the nature of the operations used to quantify an attribute. Two types of method may be distinguished:

- Subjective — quantification involving human judgment.
- Objective — quantification based on numerical rules such as counting. These rules may be implemented via human or automated means.

Figure 4 shows the specific proposed metamodel for MDWE approaches. Different levels for properties and quality characteristics are explained below:

- Feature (FT-<Level 0>): It is a general concept. A set of properties, but a higher-level concept of an approach, that broadly describes it. They may be, for example, the software development process, MDE aspects, Web Modeling aspects, tool support, experience of an approach or quality assurance techniques, among others. A Feature has a set of Sub-Features.
- Sub-Feature (FT-<Level 1>): It is a specific concept. A set of properties, but a lower-level concept of an approach. For example, the MDE Feature may have some Sub-Features such as, Language Definition, Transformations and Trace Generation. It is used to categorize the approach properties in two levels (Feature and Sub-Feature).
- Property: It should indicate the degree to which a Sub-Feature is measured. A property is used for describing and analyzing Sub-Features. In terms of properties, the aim is to look for a series of qualitative and quantitative properties based on their nature, although it might be interesting to have standard properties on MDWE that would be, somehow, centralized. In the literature, numerous references to metrics can be found, however, standardization has yet to be carried out. Furthermore, the metrics used must be validated theoretically or empirically. The theoretical validation helps know when and how to apply metrics and the empirical validation intends to prove the practical application of the proposed properties.

A base property may be defined as a measure that provides the degree of implementation of a need to be covered, whereas a derived property may be defined as a property that is composed of a set of base properties or derived properties. MDWE approaches will be evaluated in this way so most of the methods are going to be subjective since they involve human judgment. With regard to properties, it must be mentioned that a Property describes a MDWE approach element, therefore it could be considered a simple metric that indicates the state of the implementation of this element. Measurement is the act of determining a measure. The IEEE Standard Glossary of Software Engineering Terms [16] defines metrics as "a quantitative measure of the degree to which a system, component, or process possesses a given attribute".

Apart from that, as explained in Dominguez-Mayo et Al. [10], quality characteristics (hierarchically structured by Quality Characteristics (or QC-<Level 0>) and Quality Sub-Characteristics (or QC-<Level 1>)) are quality aspects, together with those properties that authors have

to assure to development teams. For instance, such an assurance could be Usability, Functionality, Maintainability, Reliability and Portability.

As shown in figure 4, authors should define relations between these properties and quality characteristics in order to identify how each Sub-Feature is influenced by each Quality Sub-Characteristic. These association links would represent the dependencies between properties and quality characteristics. They would show quality characteristics affected by Sub-Features or areas of the approach that will be significantly affected, if the approach changes.

Association links may be based on proven and real-world experience. The impact of each Sub-Feature on Quality Sub-Characteristics must be demonstrated and the requirements must be determined by real case studies applied to a number of real projects. This should be supplemented by references to published literature. A weight is used to define the importance of a property in the value of a Sub-Feature. This description or checklist can help authors discover tradeoffs and weak? points of a MDWE approach. For instance, Web Modeling is described as a Feature that has different Sub-Features. These Sub-Features could be Web Conceptual Levels, Development Process and some Web conceptual levels such as, Content Modeling, Presentation Modeling or Navigation Modeling, among others. In addition, Usability can be described as a Quality Characteristic that has different Quality Sub-Characteristics, such as Learnability, Understandability or Attractiveness, among others. Then, depending on experiments and real-world experience some aforementioned Sub-Features could be associated with previously mentioned Quality Sub-Characteristics. For example, Presentation Modeling could be related to Attractiveness and, the use of a standard like UML for modeling web applications is a property influencing Usability. The relationships between properties and quality characteristics can be defined in terms of development teams and authors' knowledge and experience.

Those quality characteristics and properties can be evaluated as elements of a MDWE approach from either a general or a specific granularity. For that reason, there are two levels of quality characteristics and properties which have, in turn, been divided into two levels: firstly, Quality Characteristics and Sub-Characteristics for quality characteristics and, secondly Features and Sub-Features for properties. In the same way, authors could have different viewpoints on the influence of properties on quality characteristics.

Thus, a generic and basic metamodel, which categorizes the concepts of this domain in a generic and easy way, is proposed. In this sense, the present proposal consists in capturing these properties by means of checklists. Checklists, which include the common elements of all these approaches, have been defined. As analyzed before, a set of properties (organized in two levels, Features and Sub-Features) that lays the basis for specifying the checklists to analyze MDWE approaches, are described and explained. For those tasks, a Systematic Literature Review (SLR) process dealing with the observation of trends and technology on the rise has been carried out. This SLR is further explained in next sections. In addition, development teams have to identify the importance of these elements considering the removal or insertion of more quality characteristics, depending on the current scope captured. Then, a Group Decision Making (GDM) is performed [13] as a decision-making problem for reaching a consensual view. The idea consists in facilitating a consensual reaching process when evaluating properties.

4 Characterization of Model-Driven Web Engineering development approaches

The main purpose of the characterization is to be able to analyse and identify tradeoffs and sensitive points of the approach studied. The aim is to evaluate and determine which properties need to be improved on a MDWE approach. In order to carry out the analysis and evaluation of MDWE approaches, the framework would use the checklists results with the implemented properties. The checklists have been drawn from a thorough systematic study of the literature and state-of-the-art of current domains. Checklists are a complete description of all properties that a MDWE approach can implement. Authors have to identify these needs and focus them on MDWE approaches quality enhancement. They have to become observers and listen to development teams in order to offer them a more effective and efficient framework.

The value of every Feature and Sub-Feature is going to be calculated in terms of its individual importance. A weighted average formula that takes into account the proportional relevance of each component, rather than treating it equally is defined by default in order to calculate the values of Feature and Sub-Features. This is a first approach to the results, but it is possible to use more accurate formulas which have already been described in the literature. So, If a stakeholder considers that, during the quality analysis of MDWE approaches within their evaluation, all properties are important, then the completeness of these approaches will be evaluated. However, development teams have to determine which of these properties are more suitable to their needs and based on that decision, they have to choose which methods better match their interests.

Descriptive statistics can be helpful when describing certain characteristics of a product and a process [34, 20]. The most important descriptive statistics are measures of central tendency such as the mean, measures of variability such as the standard deviation and range, and measures of the data distribution. In this case, descriptive statistics to describe properties and quality characteristics of a MDWE approach can be used. The arithmetic average, or the mean, is a statistic formula that measures the central tendency of a set of data; knowing the central point of a set of data is highly important. To compute the mean, it is simply necessary to sum all the observations and divide them by the total number of observations. Different points of view of an approach can be obtained.

The weighted arithmetic mean (or weighted average) is used, when trying to combine average values from samples of the same population with different sample sizes. In this application, the sample sizes represent a measure for the reliability of influence some respective values have on the mean. Authors and development teams can obtain a specific and general knowledge of the approach environment for Sub-Features and Features directly from the results. Therefore, the result would be a report with the conclusions of the evaluation. It would provide an assessment report of the approach and it may even be used to compare other MDWE approach evaluations. The formulas for analyzing the approach are described below in formula (1), (2) and (3).

It is specifically interesting to obtain a relative value for MDWE approaches in order to compare these types of approaches. The global properties value r_F can also be obtained considering the weight and value of each Feature, as shown in formula (1). Where w_{F_i} represents the weight for indicating how each Feature value r_{F_i} influences the global properties value r_F . The weights may be

different for each author or development team and may be customized. m represents the number of Features associated with the global properties value r_F .

$$r_F = \frac{\sum_{i=1}^{i=m} w_{F_i} r_{F_i}}{m} \quad \textbf{(Formula 1)}$$

- r_F : Global properties value
- w_{F_i} : Feature weight for the Feature i.
- r_{F_i} : Feature value for the Feature i.

Quality of approaches [10] in turn depends on some Features of properties, such as the MDE, the knowledge of MDWE approach development teams, the Web Modeling, the customization Modeling, the maturity of a MDWE approach and the tool support used for Modeling and transformations. Development teams and authors of an approach apply the available Modeling languages, tools and processes and develop models based on their knowledge of the problem and their experience. Besides, a Sub-Feature could influence a Feature differently and properties could also influence Sub-Features in a different way.

For Feature values the formula is formula (2). $w_{f_{i,j}}$ represents the weight indicating how each Sub-Feature value $r_{f_{i,j}}$ influences the associated Feature r_{F_i} . The weight $w_{f_{i,j}}$ may be also customized by authors and development teams. Finally, n stands for the number of Sub-Features associated with the Feature r_{F_i} , as shown in formula (2).

$$r_{F_i} = \frac{\sum_{j=1}^{j=n} w_{f_{i,j}} r_{f_{i,j}}}{n} \quad \textbf{(Formula 2)}$$

- r_{F_i} : Feature value i.
- $w_{f_{i,j}}$: Sub-Feature weight for the Feature i.
- $r_{f_{i,j}}$:Sub-Feature value for the Feature i.

Finally, for Sub-Feature values, a Sub-Feature value $r_{f_{i,j}}$ is calculated as:

$$r_{f_{i,j}} = \frac{\sum_{k=1}^{t} w_{m_{i,j,k}} r_{m_{i,j,k}}}{t}$$ **(Formula 3)**

- $r_{f_{i,j}}$: Sub-Feature value j for the Feature i.
- $w_{m_{i,j,k}}$ Property weight for the property k, sub-Feature j and Feature i.
- $r_{m_{i,j,k}}$: Property value for the property k, sub-Feature j and Feature i.

where $w_{m_{i,j,k}}$ represents the weight indicating how each property value $r_{m_{i,j,k}}$ influences the Sub-Feature $r_{f_{i,j}}$ associated. The weights may be different for each author or development team and may be customized. t represents the number of properties associated with the Sub-Feature value $r_{f_{i,j}}$.

4.1 Identification of properties

The definition of properties involves a large initial effort. Furthermore, in this approach, the description domain is built by conducting an initial description of the domain according to the information gathered from different experts, authors and development teams of MDWE approaches and by carrying out a Systematic Literature Review (SLR) process. Then, it is important to distinguish concepts such as problem analysis and decision-making for they are completely different. The SLR process must be firstly performed in order to determine the initial description of the domain. After that, the information gathered in that process may be addressed to decision-making process.

In order to get a common description of these properties, a consensus reaching process has to be defined. Group Decision-Making (GDM) methods are the central axis of other papers [13] to customize a set of weight values related to properties, as well as quality characteristics of approaches in terms of the importance given. With this method, a consensual decision for the weight values of the quality model elements can be reached. Similarly, a GDM method should be defined in the future to agree on the description of properties.

In table 1 a pattern is defined to build each Feature. All fields are relevant, but the version field is really significant, for some information is needed to control the current state of the checklists based on the quality model. Since technology is constantly changing and evolving, authors and development teams of MDWE approaches have to continuously update this first proposed quality model for MDWE approaches. In other words, it is important to take into account that this first identification is just a first base to start evaluating MDWE approaches quality. It must be assured that when analyzing several MDWE approaches, the analysis is performed with the same version of checklists.

Table 1. Pattern to describe the Features

FT-<ID>	**Name**	**Description**	**Version**	**Sub-Features**
<id value>	*<Feature name>*	<Feature description>	*<version number>*	• *SF-<ID>: <Sub-Feature name>* • …

- FT<ID>: It represents an identification code for the Feature.
- Name: It refers to the name of the Feature.
- Description: It provides a brief description of the Feature.
- Version: It consists in a number given to control the version of the Feature.
- Sub-Features: It refers to the set of Sub-Features belonging to the Feature. Each line consists of the identification value and the Sub-Feature name.

The pattern for defining each Sub-Feature is described in Table 2.

Table 2. Pattern to describe the Sub-Features

SF-<ID>	**Name**	**Description**	**Properties**
<id value>	*<Sub-Feature name>*	*<Sub-Feature description>*	• *PD-<ID>:<Property name>* • ...

- SF<ID>: It represents an identification code for the Sub-Feature.
- Name: It refers to the name of the Sub-Feature.
- Description: It provides a brief description of the Sub-Feature.
- Properties: It refers to the set of properties belonging to the Sub-Feature. Each line consists of the identification value and the property name.

Additionally, the pattern for defining the scale of Properties is described in Table 3:

Table 3. Pattern to define the scale of a Property

SC-<ID>	**Type**	**Range**	**Range type**	**Value**
<id value>	*<Quantitative or Qualitative>*	*<Range of possible values>*	*<Range type>*	*<Normalized quantitative value>*

- SC<ID>: It represents an identification code for the scale type.
- Type: It is the scale type that can be either Quantitative or Qualitative.
- Range: It is used for defining the possible values of this scale.
- Range type: It is the element type of the Range values.
- Value: It is a normalized quantitative value in terms of range.

In order to define properties, the pattern determining the checklists of Properties is described together with the elements in Table 4, where each property entity is explained as a regular expression and "+" indicates that there is one or more of the preceding elements and "|"represents the operator "or".

Table 4. Pattern to define the checklists of Properties

PD<ID> Property Name	*PB-<ID> <Property name>*	*<Property description>*		*<Reference to Scale Value>*
	PD-<ID> <Property name>	*PB-<ID> <Property name>*	*<Property description>*	*<Reference to Scale Value>*
		PD-<ID> <Property name>	…	

- PD-<ID>: It is a derived property. It consists of one or more base properties. The ID is an identification code for the property. It has also a name and a version number
- <Property name>, {PB<ID> | PD<ID>}+
- PB-<ID>: It is a base property. It includes a description defining the property and a reference to the scale value used for measuring the description. <Property name>, <Property Description>, <Reference to Scale Value>
    - <Property name>: It is the name of the property.
    - <Property Description>: It is a sentence that questions whether an approach has an element or not.
    - <Reference to Scale Value>: It is the identification code for the scale value. It is possible to have more than one reference for a scale value in this field, for example, one reference to indicate a quantitative value and another one to indicate a qualitative value. This happens because it could be necessary to indicate a quantitative value and the way it has been supported by a qualitative value.

4.2 Systematic Literature Review process for properties

One of the most important tasks to describe the specific environment is to clearly delimit the scope of the Features, Sub-Features and properties of approaches that are relevant for the study. This is obtained through the SLR (Systematic Literature Review) process [21, 22]. A problem regarding this survey is how to describe each Feature, Sub-Feature and property of approaches in a homogeneous way and how to compare them. This description is found in the next section where different patterns for Features, Sub-Features and properties are defined.

As introduced, Features, Sub-Features and properties to describe approaches should fulfill and must be consistent with the work of this survey: to improve the analysis of MDWE approaches based on a characterization of MDWE approaches (its characterization). Thus, following the terminology described in previous sections for a particular development of this review, Context, Objectives, Methods and Results are defined:

- Context: It is the systematic or even automatic analysis of MDWE approaches according to a characterization of this kind of approach. To improve the analysis of approaches using

checklists could be a solution. This study is focused on MDWE approaches. The use of these checklists makes the analysis and the evaluation of approaches easier.

- Objectives: After defining the context, the objectives of this overview have to be defined. They can be categorized into in four groups:
  - The ones dealing with identifying comparative studies that have been proposed to address the analysis of MDWE approaches.
  - Those that analyze if they offer a set of suitable characterization that can be used to analyze MDWE approaches.
  - The ones that identify the gaps in current research.
  - Those that propose future work dealing with the comparative studies of MDWE approaches.
- Methods: The search strategy for the review was centered on three lines:
  - It was primarily directed towards finding published surveys that compare MDWE approaches. Some previous surveys were found [27, 39, 12, 37]. This search started with the concepts included in these papers and then an analysis of which of them covered this domain was carried out. The proposed study of Schwinger et al. was a good basis for this framework, but not enough because this is a static study and we not only want to evaluate proposals, but carrying out a complete quality management about this domain in future. In addition, in a good strategy, the properties that are shown in the study are not enough since in a quality continuous improvement past, present and future trends have to be considered. All this favors a good model design in order to conduct a quality management, but based on a quality continuous improvement.
  - After that, a Web-search was performed to find other relevant and new concepts related to these properties. Several sets of keywords were used by combining the concepts of this study such as: “Model-Driven Engineering, Web Modeling, Model-Driven Web Engineering”, “Navigation Modeling” or “Content Modeling”, among others. These sets were used both in specialized search engines and in general ones. The search engines were: Google, Google Scholar, Scoupus, EI Compendex, ISI Web of Knowledge, IEEEXplore, ACM Digital Library and CiteSeerX
  - The last step consisted in looking at references of papers taken from previous reviews.
- Results: The final result of searching an initial description of MDWE approaches properties (including all the references from previous comparative surveys and approaches found) was translated into the analysis of various MDWE approaches.

4.3 Initial description of properties

Relating to the results obtained in the SLR process, a set of Features, Sub-Features and properties identifying MDWE properties, has been identified, classified and described regarding work and the current literature. A SLR process aims at providing an exhaustive summary of the relevant literature to a research question; which are the entities involved in a MDWE approach that best describes it? The

first step of the SLR process consisted in a thorough literature search, for example, in electronic resources such as Google Scholar, Web of Science, ScienceDirect, PubMed and Public Library of Science. The SLR process intends to provide a detailed summary of literature relevant to this research question. Next, the titles and the abstracts of the identified articles were checked for eligibility and to improve the search. Third, a list of appraisal criteria was applied to the selected articles. These criteria had to do with the methodological quality of the studies, relevance and credibility that could be implicit to the results. The Feature pattern table with their specific values for this domain is described in Table 5. Web Modeling, MDE, Experience and Tool Support Features are described in the table together with their Sub-Features.

Table 5. Features

FT -<ID>	Name	Description	Version	Sub-Features
1	*Web Modeling*	*It describes the specific MDWE aspects. It covers aspects such as evaluating the Web application development process, specific conceptual levels for this domain and levels of abstraction which have been defined in the approach. The MDE Feature is more general than this one.*	*2.1*	• *SF-<11>: Web Conceptual Levels* • *SF-<12>: Interfaces* • *SF-<13>: Development Process* • *SF-<14>: Content Modeling* • *SF-<15>: Presentation Modeling* • *SF-<16>: Navigation Modeling* • *SF-<17>: Business Modeling*
2	*MDE*	*It describes the MDE aspects as the Modeling language definition used, such as their suitability for the MDWE domain, complexity, transformations, traces, test cases, and rule generation models as a prerequisite for successfully employing MDE in the style of the MDA of the OMG.*	*2.1*	• *SF-<21>: Levels of Abstraction* • *SF-<22>: Standard Definition* • *SF-<23>: Model-Based Testing* • *SF-<24>:Transformations* • *SF-<25>:Traces*
3	*Experience*	*It describes the state of being mature in a methodology building process by authors. For example, it deals with the year of introduction of the approach, number of Modeling examples or number of applications in Real-World projects.*	*2.1*	• *SF-<31>: Topicality* • *SF-<32>: Modeling Examples* • *SF-<33>: Application in Real-World Projects* • *SF-<34>: Publications* • *SF-<35>: External Web References*
4	*Tool Support*	*It is used for specifying whether it provides or not a tool support such as a creation tool, edition tool or other different tools it supports.*	*2.1*	• *SF-<41>: Creation, edition and composition tool support* • *SF-<42>: Analysis Tool support* • *SF-<43>: Transformation Tool support* • *SF-<44>: Code generation and specific platform tool support* • *SF-<45>: Trace Tool support* • *SF-<46>: Teamwork tool support*

The Sub-Feature pattern table of FT-<1>: Web Modeling Feature with their values is described in Table 6 together with properties.

Table 6. Sub-Features (of the Feature FT-<1>: Web Modeling)

SF-<ID>	Name	Description	Properties
11	*Web Conceptual Levels*	*It describes which Web conceptual levels and which levels of abstraction are considered by an approach.*	• *PD-<111>: Content (Content Model)* • *PD-<112>: Presentation (Presentation Model)* • *PD-<113>: Navigation (Navigation Model)* • *PD-<114>: Business (Process Model, Development Team Model and Context Model)*
12	*Interfaces*	*It describes how the interrelationships among the Web conceptual levels are modeled.*	• *PD-<121>: Interface specification*
13	*Develo pment Process*	*It describes the development process and whether it is defined or not in the approach.*	• *PD-<131>: Development*
14	*Content Modeling*	*This Web Conceptual Level is described for each level of abstraction, but it focuses on the content aspects.*	• *PD-<141>: CIM* • *PD-<142>: PIM* • *PD-<143>: PSM*
15	*Presentation Modeling*	*This Web Conceptual Level is described for each level of abstraction, but it focuses on the presentation aspects.*	• *PD-<151>: CIM* • *PD-<152>: PIM* • *PD-<153>: PSM*
16	*Navigation Modeling*	*This Web Conceptual Level is described for each level of abstraction, but it focuses on the navigation aspects.*	• *PD-<161>: CIM* • *PD-<162>: PIM* • *PD-<163>: PSM*
17	*Business Modeling*	*This Web Conceptual Level is described for each level of abstraction and it is used to describe the context, the development team and the business process of a Web application.*	• *PD-<171>: CIM* • *PD-<172>: PIM* • *PD-<173>: PSM*

The Sub-Feature pattern table of FT-<2>: MDE Feature with their values is described in Table 7 together with properties.

Table 7. Sub-Features (of the Feature FT-<2>: MDE)

SF-<ID>	Name	Description	Properties
21	*Levels of Abstraction*	*It describes properties to indicate the Level of Abstraction (CIM, PIM, PSM and Code) which are used in the approach*	• *PD-<211>: CIM* • *PD-<212>: PIM* • *PD-<213>: PSM* • *PD-<214>: Code*
22	*Standard Definition*	*It describes properties for defining whether is made with standards notations or is not made. This Sub-Feature is for the evaluation of whether a web modelling language has been defined explicitly in terms of a metamodel (including UML profiles), a grammar, a semantic description in terms of semantic web technologies, or if such a definition is absent*	• *PD-<221>: Metamodel, Schema, Grammar or Ontology* • *PD-<222>: Model or Visual Syntax*
23	*Model-Based Testing*	*It describes properties for describing whether a Model-Based Testing is defined for the approach*	• *PD-<231>: Metamodel, Schema, Grammar or Ontology for Test Cases* • *PD-<232>: Model or Visual Syntax for Test Cases* • *PD-<233>: Transformations for Model-Based Testing*

24	*Transformations*	*The Transformations Sub-Feature is for the evaluation of whether approaches might support or not support various types of model transformations. For example, an approach might support transformations between platform-independent models (PIM2PIM), and transformations between platform-independent and platform-specific models (PIM2PSM), transformations between platform-specific models and code (PSM2Code)*	• *PD-<241>: Transformation Types* • *PD-<242>: Model-Driven Reverse Engineering or Synchronization*
25	*Traces*	*The Traces Sub-Feature evaluates if a generation of traces has been defined from transformations or between models. Regarding MDE, the traceability mechanism links elements of different models in order to specify elements useful in generating others. Those links can also be used to analyze impacts of model evolutions onto other models in the transformation chain*	• *PD-<251>: Trace Generation Language* • *PD-<252>: Horizontal Trace Generation* • *PD-<253>: Vertical Trace Generation*

The Sub-Feature pattern table of FT-<3>: Experience Feature with their values is described in Table 8 together with properties.

Table 8. Sub-Features (of the Feature FT-<3>: Experience)

SF-<ID>	**Name**	**Description**	**Properties**
31	*Topicality*	*The Topicality Sub-Feature is the year of introduction of the approach and other issues related with the time under development.*	• *PD-<311>:Years of Experience*
32	*Modeling Examples*	*The Modeling Examples Sub-Feature is the number of different and existing modelling examples and their depth which would also be of interest. Such a depth measure could be composed of the number of modeling concepts used, i.e. the number of content classes, nodes, links, etc*	• *PD-<321>: Experience in Examples*
33	*Application in Real-World Projects*	*The Application in Real-World Projects Sub-Feature describes the employment in designing real-world applications and number of organizations which are currently using the approach or an adaptation of it. This criterion evaluates whether real-world applications exist or do not exist.*	• *PD-<331>: Experience in Projects*
34	*Publications*	*The Publications Sub-Feature describes the number of publications in different conference proceedings, journals, books, etc*	• *PD-<341>: Experience in Publications*
35	*External Web References*	*The External Web References Sub-Feature describes the number of external web references which may be used for future development teams of the approach. For example it could be the number of references showed by a google search*	• *PD-<3511>: Number of external web references on Google* • *PD-<3512>: Number of external web references on Google Scholar*

Feature pattern table of FT-<4>: Tool Support Feature with their values is described in Table 9 together with properties.

Table 10 describes a scale property with two types of values, qualitative and quantitative, as an example.

Table 9. Sub-Features (of the Feature FT-<4>: Tool Support)

SF-<ID>	Name	Description	Properties
41	*Creation, edition and composition tool support*	*The Creation, Edition and Composition Tool Support Sub-Feature is to describe properties for analyzing the aspects of a tool support used to the creation, edition and composition of metamodels and models*	• *PD-<411>: Creation, edition and composition of Models* • *PD-<412>: Creation, edition and composition of Models for Testing* • *PD-<413>: Pattern design Tool for Models*
42	*Analysis Tool support*	*The Analysis Tool Support Sub-Feature describes properties for analyzing the features of an analysis tool support used to analyze models, metamodels and transformations*	• *PD-<421>: Analysis of Models* • *PD-<422>: Transformation Rules* • *PD-<423>: Model-Based Testing* • *PD-<424>: Trace* • *PD-<425>: Metrics*
43	*Transformation Tool support*	*The Transformation Tool Support Sub-Feature specifies properties for analyzing the aspects of a transformation tool support used to define transformations between models*	• *PD-<431>: Transformation Types* • *PD-<432>: Model-Driven Reverse Engineering or Synchronization*
44	*Code generation and specific platform tool support*	*The Code Generation and Specific Platform Tool Support Sub-Feature specify properties used to analyze the value and features related to a tool support for generating code*	• *PD-<441>: Language or Platform models and code generation* • *PD-<442>: Data Persistence models and code generation* • *PD-<443>: Web services, BPEL and Mashups models and code generation* • *PD-<444>: Models and Code for Testing* • *PD-<445>: Web 2.0 and Rich Internet Application models and code generation* • *PD-<453>: Web 3.0 and Semantic Web models and code generation*
45	*Trace Tool support*	*The Trace Tool Support Sub-Feature describes properties for analyzing the value and features of a trace tool support which is used for tracing between models and metamodels*	• *PD-<451>: Trace Generation Tool* • *PD-<452>: Horizontal Trace Generation Tool* • *PD-<453>: Vertical Trace Generation Tool*
46	*Team work tool support*	*The Team Work Tool Support Sub-Feature describes properties for analyzing the value and features of a Team work tool support. This tool is used for improving the work in a team*	• *PD-<461>: Team work*

Table 10. Property scale

SC-<ID>	Type	Range	Range type	Value
1	*Qualitative*	*{Not Supported, Partly Supported, Supported}*	*STRING*	*{0, 1/2, 1}*
2	*Quantitative*	*{0 - 5}*	*INTEGER*	$\frac{\text{Value in the Range}}{\text{MAX value in the Range}}$

4.4 The Checklists

Table 11 Checklist for the SF-<14>: Content Modeling

MT-<141> CIM 2.1	MB-<1211> CIM-Requirements Capture	It determines a well-defined technique, process and artifact used for capturing content requirements		ValueOf(SC-<1>)
	MB-<1212> CIM-Content Requirements Definition	In CIM level of Abstraction for defining and Modeling the Content it uses standard diagrams based on UML, UML stereotypes for the specific Web concepts or BPMN for the Business Process Diagram specification such as class diagrams, object diagram, or other type of standard very close to it.		ValueOf(SC-<1>)
	MB-<1212> CIM-Requirements Validation	It determines a well-defined technique, process and artifact used for validating content requirements		ValueOf(SC-<1>)
MT-<142> PIM 2.1	MB-<1421> PIM–Content Analysis	In PIM level of Abstraction for Modeling the Content it uses standard diagrams based on UML, UML stereotypes for the specific Web concepts or BPMN for the Business Process Diagram specification		ValueOf(SC-<1>)
	MD-<1422> PIM – Design Pattern for Content Models	MB-<14221> Catalog of patterns	It provides users with a proven catalogue of archetype patterns: high-value model components that can be easily incorporated into UML and BPMN models.	ValueOf(SC-<1>)
		MB-<14222> MVC pattern	It uses the content model in a MVC (Model-View-Controller Pattern) architectural pattern.	ValueOf(SC-<1>)
MT-<143>PSM2.1	MB-<1421> PSM – Content Design	In PSM the level of Abstraction for Modeling the Content uses standard diagrams based on UML, UML stereotypes for the specific Web concepts or BPMN for the Business Process Diagram specification.		ValueOf(SC-<1>)
	MD-<1422> PSM - Technology and Specific Platforms	MB-<14221> Technology	It supports technology such as Ruby on Rails, Strut, Spring, JSF, or other language or Platform to design the content model in a MVC (Model-View-Controller Pattern) architectural pattern	ValueOf(SC-<1>)
		MB-<14222> Languages or Platforms	Number of languages or platforms which can be used to design the content model such as Ruby on Rails (Ruby), Ajax Framework , Struts (Java /J2EE), Spring (Java /J2EE), JSF (Java /J2EE), Catalyst (Perl), Web2py (Python), KumbiaPHP (PHP), MonoRail (.NET), Spring .NET (.NET) or ASP .NET (.NET) (HTML).	ValueOf(SC-<2>)
	PD-<1433> PSM - Data Persistence	PB-<14331> Modeling of database	It supports Modeling of database schema design and automatic generation of scripts such as DB2, MS SQL Server, MySQL or Oracle, for example.	ValueOf(SC-<1>)
		PB-<14332> Platforms	Number of software which can be used to design the object-relational mapping such as Hibernate, OpenJPA, ADO .NET Entity Framework, TopLink, JPA, EclipseLink.	ValueOf(SC-<2>)
	PB-<1424> PSM - Web 2.0 and Rich Internet Applications	It supports Rich Internet application framework (for example AJAX or GWT) for scripting which (according to W3C) can make Web pages more dynamic. For example, without reloading a new version of a page, it may allow modifications to the content of that page, or allow content to be added to or sent from that page. The former has been called DHTML (Dynamic HTML), and the latter AJAX (Asynchronous JavaScript and XML).		ValueOf(SC-<1>)
	PB-<1425> PSM -Web 3.0 and Semantic Web	It supports semantic Web technology such as Resource Description Framework (RDF), a variety of data interchange formats (e.g. RDF/XML, N3, Turtle, N-Triples), and notations such as RDF Schema (RDFS) and the Web Ontology Language (OWL), all of which are intended to provide a formal description of concepts, terms, and relationships within a given knowledge domain. (W3C recommendation)		ValueOf(SC-<1>)

It has already been mentioned that checklists help identify strengths and weakenesses of MDWE approaches. Besides, checklists are powerful artifacts to assure quality and they can be used to analyze and control the state of MDWE approaches. Consequently, authors of approaches can use these checklists as mechanisms to analyze and control the state of their MDWE approaches. Besides that, they can also be used to indicate what aspect is important within a domain. Furthermore, development teams can also use them as a mechanism to point out what they really matter. These checklists can be defined in terms of the properties description already established.

For length reasons, this paper does not show all the Checklists to analyze the methodology. The checklist Table of SF-<14>: Content Modeling sub-feature is described in Table 11 as an example of metrics definition for a sub-feature.

5 Example application: studying a visual description of NDT methodology

NDT (Navigational Development Techniques) is a methodological approach oriented to Web Engineering. In the last few years, several Web approaches were defined; OOHDM, UWE, WebML, RUX-Method or OOH4RIA, are only some examples. However, comparative studies conclude that these approaches mainly focus on the analysis and design phases and there is an important gap in Web requirements treatment. NDT tries to fill this gap. Thus, it deals with the requirements and the analysis phases. Nowadays, NDT has evolved in the enterprise environment and it covers the complete life cycle of a software project. With the use of NDT-Suite, NDT offers tool support for each phase of the life cycle.

The entire proposed approach has been applied to the NDT methodology. For length reasons, this paper does not show all the Checklists that have been used to analyze the methodology. We have analyzed the methodology considering that all Properties are equally important. Thus, the completeness of NDT is examined in order to identify the weaknesses of the methodology. However, these Properties are the Properties that include most of MDWE existing methodologies. For this reason and because of the immaturity of the domain under study we have considered each and every one of these Properties.

5.1 Analyzing the Web Modeling Feature Checklist for NDT methodology

The Web Modeling Feature Checklist has been applied to NDT methodology in order to analyze the Web Modeling domain aspects of the methodology. Nevertheless, to restrict the length of the content of this paper, only the Content Modeling Sub-Feature of the Web Modeling Feature is described in Table 12, as an example of the analysis and its later evaluation. This Sub-Feature table is one of the Checklists which describe the Web Modeling Feature in a MDWE methodology. In the example, the Checklist has values for NDT methodology.

In Fig. 5, every Sub-Feature values for NDT methodology for the Web Modeling Features is shown in the chart. In this figure, the grey line represents Sub-Feature values for an ideal methodology and the black one represents Sub-Feature values for NDT methodology.

Table 12 Filling Checklist for the SF-<14>: Content Modeling

PT-<141> CIM 2.1	PB-<1411> CIM-Requirements Capture	It determines a well-defined technique, process and artifact used for capturing content requirements		Supported
	PB-<1412> CIM-Content Requirements Definition	In CIM level of Abstraction for defining and Modeling the Content it uses standard diagrams based on UML, UML stereotypes for the specific Web concepts or BPMN for the Business Process Diagram specification such as class diagrams, object diagram, or other type of standard o very close to it.		Supported
	PB-<1413> CIM-Requirements Validation	It defines a well-defined technique, process and artifact used validating content requirements		Supported
PT-<142> PIM 2.1	PB-<1421> PIM– Content Analysis	In PIM level of Abstraction for Modeling the Content it uses standard diagrams based on UML, UML stereotypes for the specific Web concepts or BPMN for the Business Process Diagram specification		Supported
	PD-<1422> PIM – Design Pattern for Content Models	PB-<14221> Catalog of patterns	It provides development teams with a proven catalogue of archetype patterns: high-value model components that can be easily incorporated into UML and BPMN models.	Supported
		PB-<14222> MVC pattern	It uses the content model in a MVC (Model-View-Controller Pattern) architectural pattern	Supported
PT-<143> PSM 2.1	PB-<1431> PSM – Content Design	In PSM level of Abstraction for Modeling the Content it uses standard diagrams based on UML, UML stereotypes for the specific Web concepts or BPMN for the Business Process Diagram specification		Supported
	PD-<1432> PSM - Technology and Specific Platforms	PB-<14321> Technology	It supports technology such as Ruby on Rails, Strut, Spring, JSF, or other language or Platform to design the content model in a MVC (Model-View-Controller Pattern) architectural pattern.	Partly Supported
		PB-<14322> Languages or Platforms	Number of languages or platforms which can be used to design the content model such as Ruby on Rails (Ruby), Ajax Framework , Struts (Java /J2EE), Spring (Java /J2EE), JSF (Java /J2EE), Catalyst (Perl), Web2py (Python), KumbiaPHP (PHP), MonoRail (.NET), Spring .NET (.NET) or ASP .NET (.NET) (HTML)	1
	PD-<1433> PSM - Data Persistence	PB-<14331> Modeling of database	It supports Modeling of database schema design and automatic generation of scripts such as DB2, MS SQL Server, MySQL or Oracle, for example.	Partly Supported
		PB-<14332> Platforms	Number of software which can be used to design the object-relational mapping such as Hibernate, OpenJPA, ADO .NET Entity Framework, TopLink, JPA, EclipseLink.	1
	PB-<1424> PSM - Web 2.0 and Rich Internet Applications	It supports Rich Internet application framework (for example AJAX or GWT) for scripting which (according to W3C) can make Web pages more dynamic. For example, without reloading a new version of a page, it may allow modifications to the content of that page, or allow content to be added to or sent from that page. The former has been called DHTML (Dynamic HTML), and the latter AJAX (Asynchronous JavaScript and XML).		Not Supported
	PB-<1425> PSM -Web 3.0 and Semantic Web	It supports semantic Web technology such as Resource Description Framework (RDF), a variety of data interchange formats (e.g. RDF/XML, N3, Turtle, N-Triples), and notations such as RDF Schema (RDFS) and the Web Ontology Language (OWL), all of which are intended to provide a formal description of concepts, terms, and relationships within a given knowledge domain. (W3C recommendation)		Not Supported

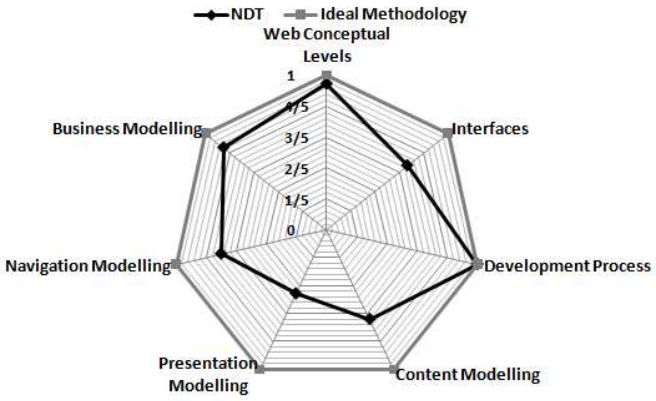

Figure 5 Web Modeling Sub-Features values

The formula 3 was applied by using the Checklist information. In this analysis, the weight value for each Property is going to be considered equally important. In fact, each weight value is 1 for each Property in order to calculate the Sub-Feature values.

Table 13 Values for calculating the SF-<14>: Content Modeling in terms of Checklists

PT-<141> CIM 2.1	PB-<1411> CIM-Requirements Capture		**1**
	PB-<1412> CIM- Content Requirements Definition		**1**
	PB-<1412> CIM- Requirements Validation		**1**
PT-<142> PIM 2.1	PB-<1421> PIM- Content Analysis		**1**
	PD-<1422> PIM- Design Pattern for Content Models	PB-<14221> Catalogue of patterns	**1**
		PB-<14222> MVC pattern	**1**
PT-<143> PSM 2.1	PB-<1431> PSM- Content Design		**1**
	PD-<1432> PSM- Technology and Specific Platforms	PB-<14321> Technology	**1 / 2**
		PB-<14322> Languages or Platforms	**1 / 5**
	PD-<1433> PSM- Data Persistence	PB-<14331> Modeling of database	**1 / 2**
		PB-<14332> Platforms	**1**
	PB-<1434> PSM- Web 2.0 and Rich Internet Applications		**0**
	PB-<1435> PSM-Web 3.0 and Semantic Web		**0**
			2/3

As far as applying formulas is concerned, the total value for each Sub-Feature is the sum of Property values associated to the Sub-Feature, divided by the number of total Properties in the Sub-Feature considered, as it is indicated in the formula 2. For qualitative value, and as it is defined in the scale SC-<1>, the Property value in the example is 1, if it is a Supported, 1/2 of the arithmetic mean of supported elements out of the total elements, if it is partly supported, and 0 if it is not supported. In case we have a quantitative value, as it is defined in scale SC-<2>, the value is out of the range max

value defined in the scale. The total value for the Content Modeling Sub-Feature is 2/3, as it is shown in Table 15.

As regards the results, in the chart, the NDT methodology has the best scores for the Development Process and the Web Conceptual Levels. Nevertheless, it may improve the Navigations Feature Modeling, Business Modeling and Content Feature Modeling Sub-Features. Therefore, the Presentation Modeling is a Web conceptual Level which this methodology lacks. So, the approach the author has to improve this Sub-Feature in the future. Another Sub-Feature which may be improved is the interfaces, which this methodology also lacks.

This analysis can help identify what the Features and Sub-Features to be improved in a methodology are. Attending to the information obtained in the Checklists, we recommend the authors to improve the Presentation Modeling Sub-Feature, since it has no models for defining either PIM or PSM levels of abstraction. Besides, it could be relevant that the models provided by the methodology in the future would be standard diagrams based on UML or UML stereotypes for the specific Web concepts. Thus, development teams would be also provided with a proven catalogue of archetype patterns in order to be helped to design the presentation.

As regards the Business Modeling Sub-Feature, it would be appropriate that the methodology may offer in the future the Business Process Execution Language (BPEL), the short version of Web Services Business Process Execution Language (WS-BPEL), which is a standard executable language for specifying interactions with Web Services. In addition, it would also be relevant that the methodology may provide technologies for the use of mashups (a way to create new Web applications by combining existing Web resources by using data and Web APIs) such as consumer mashups, data mashups or business mashups.

Furthermore, and with regards to interface Sub-Feature, the methodology should define more mechanisms used to specify the interfaces among different conceptual levels with a standard language as OCL, QVT or ATL.

In general, and in order to improve the Web Modeling Feature in NDT, we suggest to improve the PSM level of abstraction, consequently some platforms can be used to design every conceptual model such as for example Ruby on Rails (Ruby), Ajax Framework, Struts (Java /J2EE), Spring (Java /J2EE), JSF (Java /J2EE) or ASP .NET (.NET). In order to improve the methodology, we also recommend the implementation of the Rich Internet Applications and Semantic Web aspects.

With regards to the other Features, as is shown in Fig. 6, for the MDE Feature (Fig. 6 - A), NDT methodology may improve on Standard Definition and transformation Sub-Feature. Standard Definition would improve, if NDT methodology implements the standard BPMN 1.2 and 2.0 for the definition of business processes. That is, it would improve in the transformation Sub-Feature, if it provides mapping functions or transformations such as: CIM2CIM, PIM2PIM, PIM2Code and PSM2Code. Furthermore, it should also provide a synchronization method or a reverse engineering technique between transformations such as PSM2PIM and Code2PIM.

The NDT methodology does not offer a separate model for describing the transformation such as a separate Platform-Description Model, which could be very useful.

It is a methodology with good scores in Traces, Levels of Abstraction and Model-Based Testing Sub-Feature values. According to the Experience Sub-Feature, it is a methodology that lacks in experience if compared to other older proposals, for example, UWE, WebML or OOHDM methodologies. To sum up, NDT methodology has to improve in the external Web references and the number of publications (Fig. 6 - B).

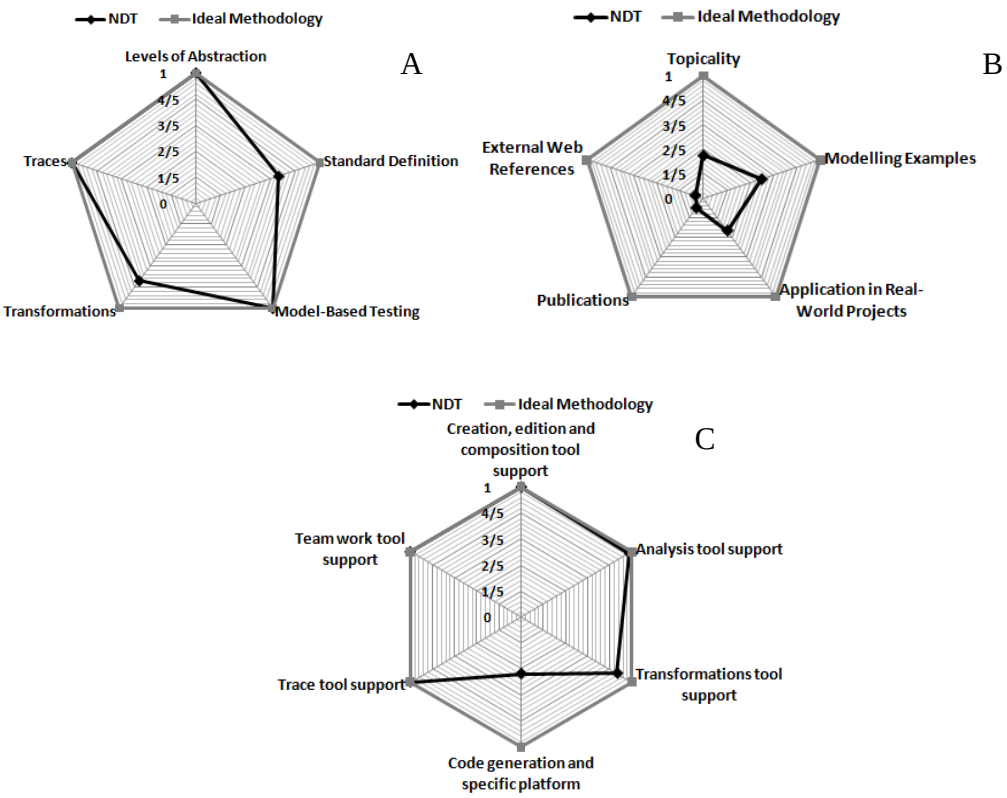

Figure 6 MDE (A), Experience (B) and Tool Support (C) Sub-Features values

As for the Tool support Feature (Fig. 6 - C), NDT methodology is supported by a complete tool. Enterprise Architect is a tool which supports the majority of aspects considered in the methodology. However, NDT methodology may improve specifically for the Transformations and code generation and specific platform Sub-Feature. The aspects that should be implemented in order to generate automatic code when using the methodology are the Web services, BPEL and Mashups models, code generation for Web Rich Internet Application models and code generation for Semantic Web models.

5.2 Global results of Features

The results for the Sub-Features of NDT have already been shown in previous sections. Nevertheless, a general view of NDT can also be shown. Fig. 8 offers the methodology results in terms of Features. The black line represents Feature values on NDT methodology and the grey one stands for Feature values for an ideal approach, depending on the Features under consideration. In this case, the total value for each Feature is the sum of the Sub-Feature values associated to the Feature (which has been obtained by the formula 2.

Table 14 Values for calculating the FT-<1>:Web Modeling in terms of the Sub-Features

SF-<ID>	Name	Values
SF-<11>	Web Conceptual Levels	**1**
SF-<12>	Interfaces	**2 / 3**
SF-<13>	Development Process	**1**
SF-<14>	Content Modeling	**2 / 3**
SF-<15>	Presentation Modeling	**1 / 2**
SF-<16>	Navigation Modeling	**5 / 7**
SF-<17>	Business Modeling	**6 / 7**
		3 / 4

Table 15 Values for calculating the Feature values.

FT -<ID>	Feature name	Values
FT-<1>	Web Modeling	**3 / 4**
FT-<2>	MDE	**7 / 8**
FT-<3>	Experience	**1 / 4**
FT-<4>	Tool Support	**7 / 8**

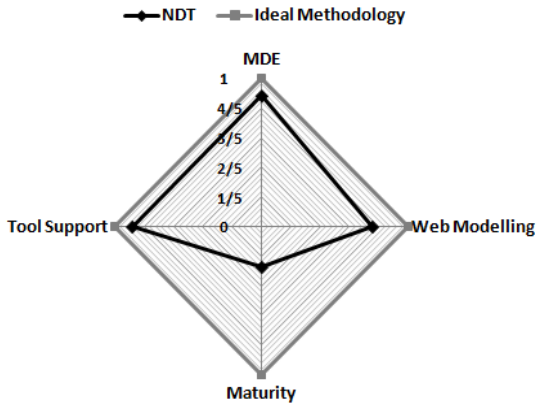

Figure 7 Features values

The NDT methodology has better results in the MDE and Tool Support Feature and a lower value in the Experience Feature. The Web Modeling Feature has also a significant value.

6 Conclusions and Future Work

In this paper, a meta-model has been defined based on ISO quality standards in order to be used with QuEF which is a framework for the quality management of MDWE methodologies. Besides, a set of patterns to instantiate the quality model has been proposed in order to make easier the instantiation of the proposed quality model. Furthermore, these patterns were used to suggest a characterization of MDWE methodologies in order to use this information for the analysis and evaluation of MDWE methodologies. The study focuses on describing Properties (hierarchical by Features and Sub-Features) for MDWE methodologies.

In QuEF, at the beginning of the Quality Model Strategy and Design phases, it is important to generally define all development teams' needs. Once all the development teams' needs are customized then, in turn, they are defined within a quality model. During the Quality Model Operation phase both, development teams and authors, can analyze, control and evaluate the quality of their approaches, as shown in the sample application included in this paper. So, a quality model has been defined for these Properties and a set of Checklists has been described in terms of this quality model. Features and Sub-Features are described in order to show a complete example of Properties.

We are currently working on the analysis and evaluation of other MDWE methodologies and even the experimentation for the validation of these Properties. So, contacts with authors of different MDWE methodologies have been established to analyze and evaluate their approaches. The idea is to get valuable feedback on the Checklists concerning understandability as well as acceptance. Besides, The filled Checklists from the development teams of the approaches could be an interesting data source for comparing different approaches as well as determining the current state-of-the-art of the whole MDWE field.

The use of a methodology or standard model process ensures consistency in their output, although it does not guarantee a product quality. This fact may even institutionalize the creation of bad products. Therefore, in the future, the product Properties in the results have to be taken into account and related to the methodology Properties because a standard process does not necessarily conclude with a quality product. In this sense, it is necessary to describe the product and define relations among the methodology Properties and the product Properties in the results. In fact, the product Properties which are obtained by means of the methodology have to be also considered and related to the methodology Properties, since it is necessary to know what methodology Properties influence product Properties. This is due to the fact that the use of a methodology does not ensure a quality product, but the uniformity of results. Thus, product Properties have to be related to methodology Properties in future works.

One of the limitations is that the proposal works with people and this is a difficult aspect to take into consideration because people have to agree on a lot of concepts. Each one has different experiences and points of view. Besides, the domain of MDWE methodologies is currently immature and these methodologies have different metamodels, models, transformations, tools support that work in a different way and some other aspects that are causing different situations that render some consideration. Take, for example, development teams who do not know how to make use of these approaches let alone know how these approaches can help them and which approach is more suitable for them; for that matter, and yet, they are the main target of design of the desired methodology. In

addition, the diversity in some features of the design of these approaches reflects that authors of these approaches do not share the same vision or purpose of use. So, this methodologies depends on the project context in which they are going to be used. The convergence towards a standardization of these methods together with a consensus within a quality management and common quality model would result in that both, authors and development teams, make use of and design these methodologies effectively and efficiently.

Cooperation among authors has to lead to the detection of needs. Authors have to provide development teams with the Properties that they need and they have to guarantee the Quality Characteristics that development teams are demanding. Thus, authors would win in effectiveness and efficiency in the design of their methodologies. This would be possible because both, development teams and authors, would have a common strategy, the same objectives as well as purposes and the same design of the model to be achieved. At the same time, and due to their shared knowledge, authors would have a clearer idea as to what should be changed in the quality model. So, this approach would be used not only for Analysis and Evaluation but by development teams of different methodologies. Development teams will have the chance to compare methodologies and authors, on their part, will have the chance to control and improve the design of their methodologies.

Acknowledgements

This research has been partially supported by the NDTQ-Framework project (TIC-5789) of Junta de Andalucía, by the TEMPROS (Testing Temprano y Modelos de Simulación Híbrida en la Producción de Software) Project of the Ministerio de Educación y Ciencia, Spain (TIN2010-20057-C03-02) and by the FEDER of European Union for financial support via project "THOT. Proyecto de innovación de la gestión documental aplicada a expedientes de contratación de servicios y obras de infraestructuras de transporte" of the "Programa Operativo FEDER de Andalucía 2007-2013". We also thank all the Staff and Researches of the Regional Government of Andalucía Agency of Public Work for their dedication and professionalism.

References

1. Abrahão, S., Condori-Fernandez, N., Olsina, L., "Defining and validating Properties for navigational models". Proceedings of the Ninth International Software Metrics Symposium (METRICS'03). pp. 200-210, 2003.
2. Abran, A., Al-Qutaish, R. E., Desharnais J.and Habra, N., Chapter 5: ISO-Based Models to Measure Software Product Quality, in: Software Quality Measurement – Concepts and Approaches, Edited by: Ravi Kumar Jain B. pp. 61-96, Publisher, Institute of Chartered Financial Analysts of India, Hyderabad, India: ICFAI University Press.2008.
3. Basili, V., "Software Modeling and Measurement: The Goal/Question/Metric Paradigm". University of Maryland, CS-TR-2956, UMIACS-TR-92-96, 1992.
4. Basili, V.R. and Weiss, D.M. A Methodology For Collecting Valid Software Engineering Data, IEEE Software Engineering ndards, Std. 610.12-1990, pp.47-48. 1993
5. Becker P., and Olsina, L., "Towards Support Processes for Web Projects", ICWE Workshops'10, pp. 102-113, 2010.
6. Cachero, C., Poels, G., Calero, C. "Towards a Quality-Aware Web Engineering Process". Twelfth International Workshop on Exploring Modelling Methods in Systems Analysis and Design. Vol. 1, pp 7-16. Held in conjunction with CAISE'07Trondheim, Norway, 2007.

7. Calero, C., Ruiz, J., Piattini, M., “Classifying Web Properties using the Web quality model”. Vol. 29, No. 3, pp. 227-248, 2005.
8. Ceri, S., Fraternali, P., and Bongio A., “Web Modeling Language (WebML): A Modeling Language for designing Web sites” Computer Networks: The International Journal of Computer and Telecommunications Networking Vol. 33, Issue 1-6, Elsevier North-Holland, NY, USA DOI: 10.1016/S1389-1286(00)00040-2, 2000
9. Domínguez-Mayo, F.J., Escalona, M.J., Mejías, M., Ross, M., Staples, G. “A Quality Management Based on the Quality Model Life Cycle”, Computer Standards and Interfaces, January 2012, ISSN 0920-5489, 10.1016/j.csi.2012.01.004.
10. Domínguez-Mayo, F.J., Escalona, M.J., Mejías, M., Ross, M., Staples, G. “Quality Evaluation for Model-Driven Web Engineering Methodologies”, Information Software Technology, Vol. 54, Issue 11, November 2012, pp. 1265–1282
11. Escalona, M.J., Aragón, G., “NDT. A Model-Driven Approach for Web Requirements”. IEEE Transactions on software engineering, Vol. 34, No. 3, pp. 377-390, 2008.
12. Escalona, M.J., Koch, N., “Requirements Engineering for Web Applications – A comparative study”. Journal of Web Engineering. Vol. 2, No. 3, pp. 193-212, 2004.
13. Espinilla, M., Domínguez-Mayo, F.J., Escalona, M., Mejías, M., Ross, M., Staples, G. “A Method Based on AHP to Define the Quality Model of QuEF”, Knowledge Engineering and Management, Proceedings of the Sixth International Conference on Intelligent Systems and Knowledge Engineering, (ISKE 2011), Publisher: Springer Berlin / Heidelberg. Vol. 123, Pág. 685 - 694. ISBN: 978-3-642-25660-8.
14. García, F., Serrano, M., Cruz-Lemus, J., Ruiz, F., Piattini, M., “Managing software process measurement: A metamodel-based approach”. Information Sciences.Vol. 177, No. 12, pp. 2570-2586, 2007.
15. Goethert, W., Fisher, M.: Deriving Enterprise-Based Measures Using the Balanced Scorecard and Goal-Driven Measurement Techniques. Software Engineering Measurement and Analysis Initiative, CMU/SEI-2003-TN-024, (2003).
16. IEEE Std 610.12-1990. IEEE Standard Glossary of Software Engineering Terminology.
17. ISO/IEC 15939:2007. Systems and software engineering -- Measurement process. Retrieved March 2013 from http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=44344
18. ISO/IEC 25040:2011. Systems and software engineering -- Systems and software Quality Requirements and Evaluation (SQuaRE) -- Evaluation process Retrieved March 2013 from http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=35765
19. ISO- International Organization for Standardization, Retrieved March, 2013, from http://www.iso.org.
20. Jones, M. B., Walsh, D., Prins, S., Kiermeier, A., Russell, K., Dialsingh, I., Govindaraju, R., Hewson, P., Gillespie, C., “Open source stage 1 statistics textbook project” Retrieved November 2012 from http://www.massey.ac.nz/~mbjones/Book/
21. Kitchenham, B.A., Hughes, R.T., Linkman, S.G.: “Modeling Software Measurement Data”. IEEE Transactions on Software Engineering, 27(9), pp. 788-804, (2001).
22. Kitchenham, B., Pearl Brereton, O., Budgen, D., Turner, M., Bailey, J., Linkman, S., “Systematic Literature Reviews in Software Engineering – A Systematic Literature Review”, Information and Software Techonology, Vol. 51, Issue No 1, pp. 7-15, 2009.
23. Koch, N., Knapp, A., Zhang, G., and Baumeister, H., UML-based Web Engineering: An Approach based on Standards (book chapter). In Web Engineering: “Modelling and Implementing Web Applications”. Gustavo Rossi, Oscar Pastor, Daniel Schwabe and Luis Olsina (Eds.), chapter 7, 157-191, 2008.

24. Maibaum, T., Wassying, A., “A Product-Focused Approach to Software Certification”. Computer, 2008. 41(2): p. 91-93
25. Meliá, S., Gómez, J., Pérez, S., Díaz, O., “A Model-Driven Development for GWT-Based Rich Internet Applications with OOH4RIA”, ICWE '08 Proceedings of the 2008 Eighth International Conference on Web Engineering, 2008.
26. Mohagheghi, P., Dehlen, V., “Developing a Quality Framework for Model-Driven Engineering”. Models in Software Engineering: Workshops and Symposia at MoDELS 2007, pp. 275–286, 2008.
27. Moreno, N., Romero, J.R., Vallecillo, A., “An overview of Model-Driven Web Engineering and the MDA”, Web engineering: modelling and implementing web applications, Human–Computer Interaction Series, 2008, Part II, 353-382, DOI: 10.1007/978-1-84628-923-1_12
28. NDT (Navigational Developtment Techniques): Retrieved February, 2011, from http://www.iwt2.org.
29. OMG: MDA. Retrieved January, 2010, from http://www.omg.org/mda/faq_mda.htm
30. OOHDM: Retrieved February, 2011, from http://www-di.inf.puc-rio.br/schwabe/HT96WWW/section1.html
31. OOH4RIA: Retrieved February, 2011, from http://code.google.com/p/ooh4ria/
32. Pérez, J. M., Ruiz, F., Piattini, M., “Model Driven Engineering Aplicado a Business Process Management”, Informe Técnico. UCLM-TSI-002, 2007.
33. Preciado, J.C., Linaje, M., Morales-Chaparro, R., Sanchez-Figueroa, F., Zhang, G., Kroiβ, C., Koch, N., “Designing Rich Internet Applications Combining UWE and RUX-Method”, ICWE '08 Proceedings of the 2008 Eighth International Conference on Web Engineering, ISBN: 978-0-7695-3261-5, 2008.
34. Reid, R. D., Sanders, N. R., “Operations Management” September 14, 2004 | ISBN-10: 0471712108
35. Rossi, G., Schwabe, D., de Lucena, C.J.P., Cowan, D.D., "An Object-Oriented Model for Designing the Human-Computer Interface of Hypermedia Applications", Proceedings of the International Workshop on Hypermedia Design (IWHD'95), Springer Verlag Workshops in Computing Series, 1995.
36. UWE (UML-Based Web Engineering): http://uwe.pst.ifi.lmu.de
37. Vallecillo, A., Koch, N., Cachero, C., Comai, S., Fraternali, P., Garrigós, I., Gómez, J., Kappel, G., Knapp, A., Matera, M., Meliá, S., Moreno, N., Pröll, B., Reiter, T., Retschitzegger, W., Rivera, J. E., Schwinger, W., Wimmer, M., and Zhang, G., “MDWEnet: A Practical Approach to Achieving Interoperability of Model-Driven Web Engineering Methods”, Proc. Third Int’l Workshop Model-Driven Web Eng., pp. 246-254, 2007.
38. WebML: http://www.webml.org
39. Schwinger, W., Retschitzegger, W., Schauerhuber, A., Kappel, G., Wimmer, M., Pröll, B., Cachero C., Castro, Casteleyn, S., De Troyer, O., Fraternali, P., Garrigos, I., Garzotto, F., Ginige, A., Houben, G-J., Koch, N., Moreno, N., Pastor, O., Paolini, P., Pelechano V., Ferragud, Rossi, G., Schwabe, D., Tisi, M., Vallecillo, A., van der Sluijs and Zhang, G., “A survey on Web modeling approaches for ubiquitous Web applications”. International Journal of Web Information Systems Vol. 4 No. 3, pp. 234-305, 2008.