

NDT-Suite: A Methodological Tool Solution in the Model-Driven Engineering Paradigm

Julián Alberto García-García, María José Escalona, Francisco José Domínguez-Mayo, Alberto Salido

Web Engineering and Early Testing Research Group, University of Seville, Seville, Spain

Email: julian.garcia@iwt2.org, mjescalona@us.es, fjdominguez@us.es, alberto.salido@iwt2.org

Received 18 February 2014; revised 15 March 2014; accepted 22 March 2014

Copyright © 2014 by authors and Scientific Research Publishing Inc.

This work is licensed under the Creative Commons Attribution International License (CC BY).

<http://creativecommons.org/licenses/by/4.0/>



Open Access

Abstract

Although the Model-Driven paradigm is being accepted in the research environment as a very useful and powerful option for effective software development, its real application in the enterprise context is still a challenge for software engineering. Several causes can be stacked out, but one of them can be the lack of tool support for the efficient application of this paradigm. This paper presents a set of tools, grouped in a suite named NDT-Suite, which under the Model-Driven paradigm offer a suitable solution for software development. These tools explore different options that this paradigm can improve such as, development, quality assurance or requirement treatment. Besides, this paper analyses how they are being successfully applied in the industry.

Keywords

Model-Driven Web Engineering, Model-Based Suite, Tools, Practical Experiences, NDT, NDT-Suite

1. Introduction

The Model-Driven Engineering (MDE) [1] paradigm consists in applying the model-driven paradigm to the software engineering context [2]. It is being considered a very attractive solution to improve software projects. Since recently, research teams have proved its benefits in the application of software development.

However, in the enterprise community context, there is still a gap regarding the application of this paradigm. Several causes can be exposed for explaining this lack, but we would like to stick out the next ones from our research experience.

Firstly, MDE deals with very abstract concepts like metamodels, transformations or constraints, among others. They are complex to be understood out of the academia, as they do not belong to software teams' "common language". Therefore, they are hardly ever trained in these concepts.

How to cite this paper: García-García, J.A., *et al.* (2014) NDT-Suite: A Methodological Tool Solution in the Model-Driven Engineering Paradigm. *Journal of Software Engineering and Applications*, 7, 206-217.
<http://dx.doi.org/10.4236/jsea.2014.74022>

Secondly, concrete syntaxes have to be developed for the application of MDE in order to instance the meta-models and concrete tools to execute transformations. Current solutions, like QVT [3] or ATL [4], can result very abstract for them. In some cases, like QVT, there is still a lack for effective tool support.

Finally, with regard to tool support, although MDE can help software products systematized or automated, as it is presented in this paper, MDE approaches offer a poor tool support which produces the lack of interest in its application.

This paper presents how this paradigm can be presented in a more friendly and efficient way to the enterprise community, softening its application and improving the application of transformations and MDE principles. The paper introduces a set of tools, grouped under a suite named NDT-Suite [5]. It demonstrates that the effective solution MDE can offer to software engineering. NDT-Suite is a solution defined for the application of NDT (Navigational Development Techniques) [2], a methodological MDE approach that has been developed in the academia and it is being applied to real contexts.

We have structured this work as follows. Firstly, Section 2 describes the related works on this topic. Section 3 briefly introduces NDT, the methodological environment where these tools can be used. Section 4 presents the main approach of the paper, NDT-Suite. After that, Section 5 points out some real project references, and the paper finishes with a set of conclusions and related work in Section 6.

2. Related Work

In the last years, the MDE paradigm has been used in different areas of software development. One of these areas is Web Engineering, which is commonly named Model-Driven Web Engineering (MDWE) [2]. Nevertheless, research groups obtained very relevant results when applying this paradigm to Web development and currently, there is a wide range of Web methodologies that belong to the MDE paradigm and define its procedure and transformations from one model to another one.

Furthermore, approaches are usually appearing with different concepts and terminologies in many cases as shown in Domínguez-Mayo *et al.* [6] [7], although all lack the use of standards and practical experience. For this reason, only the most referenced ones will be briefly described.

OOHDM [8] is a highly referenced methodology and one of the most accepted. It consists of four different design activities: Conceptual Model, Navigational Design, Abstract Interface Design and Implementation. Conceptual and navigational transformations are completely independent, except from operation invocations of conceptual PSM objects from the navigational PSM, where the type of invocation may vary. Thus, the implementation technology and platform of conceptual and navigational PSM may be selected and combine quite independently.

UWE [9] is a methodology based on Unified Modelling Language (UML) [10] for Web application development. It covers the complete Web systems development lifecycle from requirements specification to code generation, focusing on personalized or adaptive applications. The development process includes three main phases: capture of requirements, analysis and design, and implementation. The transformation rules are defined to map 1) metamodel WebRE to UWE and 2) UWE to WebRE metamodels. The first model transformation step of the UWE process deals with mapping Web requirements models to UWE functional models. The design models are: content, navigation, process, presentation and adaptation. There is a set of dependencies among these functional models that allows the creation of other models, as well as their amendments. ArgoUWE [11] and MagicUWE [12] have been developed for the computer-aided design of Web applications using UWE.

WebML [13] is a proposed methodology based on formal specification graphics in a complete design process. It is used on applications that highly interact with information. The model is divided into four phases: the process definition phase (it uses BPMN [14] notation); the data model phase, (it is defined by different data, tables or relationships and uses either graphs of Entity-Relation or UML class diagrams); the hypertext model phase (it consists in the composition model, representing hypertext sites where every site has content elements); and the navigation model, which defines the links among different sites; finally, the composition model details content items of each site. WebML is associated with a development tool called WebRatio [15], which automatically generates fully functional applications from WebML diagrams.

OOWS [16] is an extension of OO-Method. The process to define a software system for a Web environment is conducted in two steps: the conceptual model and the navigational model. The conceptual model concerns the specification of user requirements. The navigation model centers on an object model and navigation requirements using UML notation. The navigation model, in turn, is composed of a set of navigation maps (one for

each agent) to represent and structure the system global vision by defining allowable navigation. This is directly represented by means of a directed graph where nodes are navigational contexts and arches represent the navigation links. Once the navigational model is defined, the presentation characteristics are associated with the system. Presentation requirements will focus on the use of simple presentation patterns related to the different elements that conform a navigation node. The generator (compiler) will use this information stored to create diverse interfaces for each user within the Web application architecture that the OOWS method proposes.

3. NDT: Navigational Development Techniques

NDT (Navigational Development Techniques) is a model-driven Web methodology that was initially defined to deal with requirements in Web development. NDT has evolved in the last years and offers a complete support for the whole lifecycle. Today, NDT selects a set of metamodels for each development phase that allows all concepts in every phase of NDT to be modelled and formally related to other concepts by means of associations and/or OCL [17] constraints.

NDT can be characterized through the next properties:

- 1) NDT is a MDE methodology that covers the whole lifecycle (the Feasibility Study Phase, the Requirements Phase, the Analysis Phase, the Design Phase, the Implementation Phase, the Testing Phase, and finally, the Maintenance Phase), although it mainly focuses on the Requirements Phase. In this phase, NDT offers a set of techniques to capture, define and validate different kinds of requirements. These requirements are formally defined by a metamodel and they can be traced to the remaining artifacts of the lifecycle by managing them in a suitable manner. NDT uses information previously captured, defined and validated in the requirements phase as the basis for the full development of the system.
- 2) Despite its application in classical environments, NDT is implemented in relation to the Web supporting special characteristics like navigation, complex interfaces or RIA (Rich Internet Applications). In the requirements validation, NDT is oriented to cover classical techniques like traceability or prototypes, among others, but it is enriched to support these special Web characteristics.
- 3) The degree of automation of NDT is one of its more relevant qualities. As it is presented in this paper, NDT is fully supported by a set of tools grouped in NDT-Suite.

In conclusion, NDT is a theoretical approach, based on metamodels or transformations, among other elements, which is often used in companies¹. As it will be analyzed in the next sections, the tools support of the methodology entails effective support for the application of the methodology in real projects. In fact, in the last ten years, NDT has been used in a large number of real projects with a high number of companies, either public (such as the Andalusian Regional Ministry of Culture, Education and Sport, and the Andalusian Regional Ministry of Health and Social Welfare, among others) or private (for example, Airbus Military, TecnoCom, Everis, Fujitsu or Ayesa, among others). This is possible due to the fact that NDT is completely supported by a set of free Java tools (which are described in the following sections) grouped in the NDT-Suite.

4. NDT-Suite

In this section, a detailed presentation of NDT-Suite is offered.

4.1. The Global Structure

NDT-Suite works on/with a UML-based tool named Enterprise Architect (EA) [18]. We have chosen EA in order to provide a real environment to apply NDT to a business context, although it has not been an easy task. In fact, our research group (IWT2) carried out a comparative study of nine modelling tools in order to identify the best tool to perform our job regarding technology transfer projects. This study was conducted in liaison with the Andalusian Regional Ministry of Culture, but it was not published. It is worth pointing out that this study can be consulted on IWT2's website. After carrying out this comparative study, we concluded that 1) EA offers the best value for money in the evaluated aspects; 2) EA offers several important advantages, such as the possibility of defining profiles or tools for document management by drawing UML diagrams, for instance, which have been very relevant to carry out our work; and 3) EA is widely used and known by most of our customers. These reasons made us finally NDT is supported on EA.

¹A detailed list of projects where NDT is used can be consulted in the IWT2 website (www.iwt2.org) in the Project section.

Finally, **Figure 1** shows the NDT-Suite's architecture. At the lower level is Enterprise Architect. At the next level, NDT-Profile (see Section 4.2) is located. NDT-Profile is the main NDT tool because it implements each NDT's metamodel through a specific UML-profile which is integrated within EA. The remaining tools of NDT are located at the top level of the architecture.

4.2. NDT-Profile

When methodology (especially NDT) is defined with metamodels it is possible to guarantee uniformity, formalized terminology and correct definition, according to that methodology. However, if a suitable tool is not defined, the use of this methodology can result too complex and expensive. Consequently, it is required to offer a tool-based mechanism.

The OMG (Object Management Group) defines metamodel-based languages in terms of two viewpoints: to define a new language (UML alternative) or to extend UML (specializing its concepts and restricting other concepts, always keeping the original semantics of the UML elements).

We have chosen to extend UML in order to define each NDT's metamodel. This is the NDT-Profile's aim. Our reasons are: 1) UML is a recognised and widespread standard in enterprise environments; and 2) UML provides great flexibility, expressiveness and a generic extension mechanism for building UML models in particular domains (like NDT).

In this context, NDT-Profile implements within Enterprise Architect each NDT's metamodel using UML-profiles. Thanks to NDT-Profile is integrated within EA, NDT-Profile offers the chance of having all the artefacts defining NDT easily and quickly, as they are integrated within the EA tool.

To implement each NDT's metamodel, we have used the Model Driven Generation (MDG) Technologies of Enterprise Architect. MDG Technologies allow extending Enterprise Architect's modelling capabilities to specific domains and notations. In addition, MDG Technologies seamlessly plug into Enterprise Architect to provide additional toolboxes, UML profiles, patterns, templates and other modelling resources.

Finally, it is important to emphasize that all UML-profiles of NDT can be implemented by any tool which provide UML extension mechanisms. In addition, most of these tools provide import and export exchange mechanisms structured by means of XMI format.

4.3. NDT-Driver

As mentioned in Section 3, NDT is a MDWE methodology and therefore, its foundations are based on the definition of models and transformation rules among these models, which entail the production of other models.

As regards transformation rules, NDT uses QVT standard to define its derivation rules. However, QVT notations are not easy to apply in practical environments, as they do not result too friendly for development teams. Concepts such as models, metamodels, transformations or QVT, for instance, are not common notations in the enterprise environment and they seem too abstract and complex.

For these reasons, it was necessary to abstract all these theoretical concepts in order to introduce NDT in enterprise environments and make its application easier. NDT-Driver was specifically developed for this purpose. The graphical interface of NDT-Driver is very simple and intuitive. **Figure 2** shows the main interface of the tool.



Figure 1. NDT-Suite's architecture.

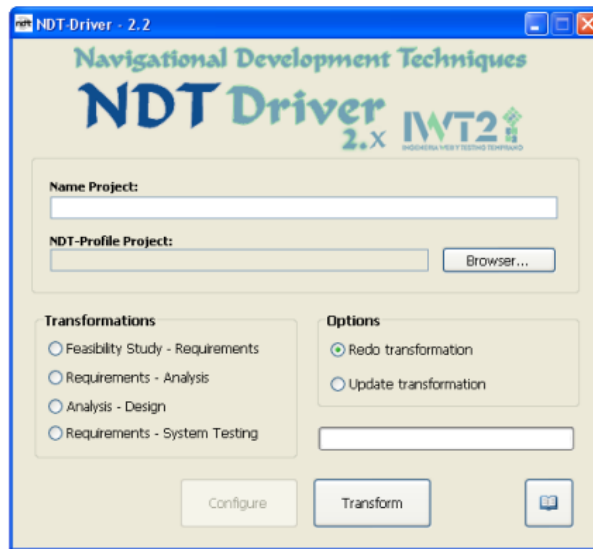


Figure 2. NDT-Driver's interface.

NDT-Driver implements a set of Java-based automatic procedures for carrying out each of the QVT transformations defined in NDT on a project specified with NDT-Profile. Furthermore, NDT-Driver can be used in projects through both, a sequential lifecycle and an evolutionary lifecycle. Models to generate can be chosen, once the transformations that must be performed have been selected.

Moreover, NDT-Driver allows choosing the model in order to support projects with evolutionary development cycles. For this, with the “Configure” button, you can select the models you want to transform. By default, all the models are selected. For instance, Figure 3 shows the modal window associated with the transformation Requirements Phase to Analysis Phase.

In addition, NDT-Driver allows selecting to *rebuilt* or *updated* (these options are showed in Figure 2) any model. For example, if the Requirements Phase is completed, the Conceptual Model² (CM) of the Analysis Phase may be generated. Then, if it is noticed that any requirement is not defined according to the user's needs, it is not totally necessary to rebuild the CM because NDT-Driver detects these changes and updates the concrete part of CM.

Finally, it is interesting to mention that NDT-Driver considerably minimizes the time spent in the design and development of models from different lifecycle phases of NDT. In fact, thanks to this tool, NDT has been successfully applied in practical environments without increasing the cost of the project.

4.4. NDT-Quality

As mentioned above, being successfully used in a large number of real projects at different companies, either public or private, is one of the main advantages of NDT and its support tools (especially NDT-Profile). This fact motivates us to implement NDT and its support tools. However, analysts could still make errors and methodological inconsistencies when defining the system, even though NDT-Profile offers a suitable environment to NDT and it manages the use of NDT artifacts and constraints. This is the reason why we have developed NDT-Quality.

NDT-Quality has a similar interface to NDT-Driver and offers a mechanism to support some of the most classical techniques for quality assurance. The relevant aspect of NDT-Quality is that it exploits the use of MDE to ease the application of this task. Thus, NDT-Quality has two main objectives.

On the one hand, NDT-Quality ensures the quality of use of NDT in every phase of the software development lifecycle. In light of this, it checks all OCL constraints of NDT metamodels and controls how analysts use NDT artifacts, *i.e.*, our tool monitors aspects such as failure to complete definitions; linguistic patterns regarding the identification of artifacts; bad definitions of constraints in artifacts; existence of cycles in class diagrams; ab-

²The Conceptual Model represents the static structure of the system.

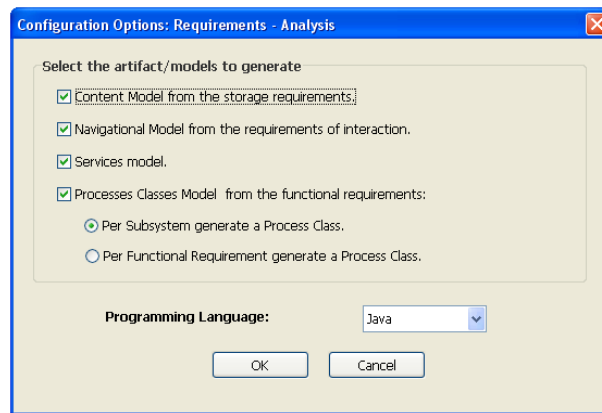


Figure 3. NDT-Driver's configuration interface.

sence of specific data in the Information Storage Requirement³ and existence of isolated Functional Requirements⁴ (*i.e.*, NDT establishes that each RF is related to an Actor or otherwise, another RF through a link “include” or a link “extend”); among other aspects.

On the other hand, NDT-Quality aims to ensure traceability among artifacts when MDE transformation rules of NDT are used. This control is necessary because NDT allows analysts to carry out transformations in order to enrich and complete their models. This aspect is crucial to ensure quality in software development framed within any methodology, and even more if this methodology is included within the Model-Driven Engineering paradigm (like the methodology NDT), for traceability is one of the pillars of this paradigm.

Finally, NDT-Quality also provides reports (in different formats) with detailed information on each identified error. In addition, NDT-Quality provides detailed information to resolve each error (see **Figure 4**).

4.5. Other MDE Solutions

The previous sections have described the main support tools of NDT, but its suite goes further. NDT-Suite is composed of other tools that cover different objectives. More information about these tools is available on IWT2 Website although these tools are briefly described below.

NDT-Prototype is a tool that generates a set of XHTML prototypes from the Navigation Models⁵ described in the Analysis Phase in a project developed with NDT-Profile. This tool is not related to the test phase, although it gives very good support for requirements validation.

NDT-Merge [18] [19] compares metamodels in order to identify syntactic and semantic inconsistencies among different versions of the same requirements catalogue.

NDT-Report is integrated into NDT-Profile and allows generating documents (in office formats) from NDT-Profile.

NDT-Glossary [20] defines and implements MDE-based protocol to generate a glossary from the Requirements Phase and establish a common glossary, which may reduce the risk of misunderstandings and facilitate communication between users and analysts.

NDT-Counter [21] measures the required effort to develop a project. This measurement is based on the use case technique.

4.6. NDTQ-Framework

The real life uncovers many problems that should not take place, although applying methodologies helps ensure quality of results. Both, elaborating documents or the correct application of methodological phases end up as mere formality in many cases. Sometimes, projects framed in a methodology experience can be delayed, changed or patched code, causing inconsistency between the documentation and the final system.

³Information Storage Requirements define what information is stored in the system and the relationships established among such information.

⁴Functional Requirements describe the needs of functionality offered by the system.

⁵The Navigation Model describes how users can navigate through the system.

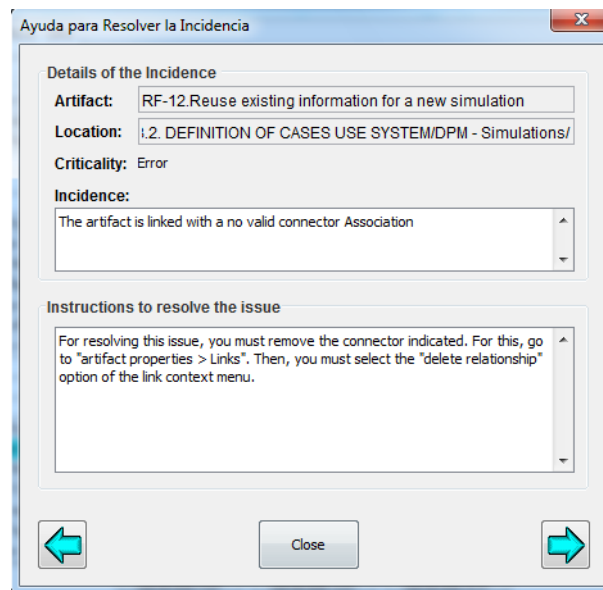


Figure 4. User interface with help information.

In consequence, we have taken out several conclusions after applying NDT in a real environment:

- 1) MDE is a good solution for software development and helps reduce cost and time, as well as increases final quality and traceability of software. However, suitable tools have to be developed, in case of being applied, as metamodels, transformations and this specific context are not friendly for development teams.
- 2) The development process is not enough for an enterprise environment. In a software project there are other aspects like quality assurance, project management or security aspects, among others, that have to be managed according to each organization's specific rules, frequently based on standards and maturity models.

For these reasons, NDT has evolved again in the last years, and now, a global framework named NDTQ-Framework is proposed, in order to offer a suitable and a global process-based solution for the real application of NDT.

NDTQ-Framework is based on a MOF [22] metamodel for Software Process Support. This metamodel is detailed in Ponce *et al.* [23] and offers an appropriate mechanism for process definition by covering both, the main software process concepts and the ISO/IEC TR 24774:2007 [24] compliant, as well as provides simplicity in order to develop a whole model-based solution.

Using an UML Profile-based metamodel on Enterprise Architect, NDTQ-Framework defines six groups of processes that are briefly introduced below:

- Software Development Processes, which are defined in terms of NDT lifecycle foundations.
- Software Maintenance Processes, which are based on the Information Technology Infrastructure Library (ITIL) [25] and Capability Maturity Model Integration (CMMI) [26]. It begins when the project is in the production stage and finishes when the system falls into disuse.
- Testing Processes, which center on the first results of ISO/IEC 29119 [27].
- Software Quality Processes, which are based on ISO 9001:2008 [28] and CMMI.
- Project Management Processes, which focus on some of the practices of Project Management Body of Knowledge (PMBOK) [29] and CMMI.
- Security Processes, which refer to ISO 27001 [30].

NDTQ-Framework is born after detecting the necessity of managing software quality assurance in an important project carried out since 2004 in liaison with the Andalusian Regional Cultural and Sport Ministry (Andalusia, Spain). However, a problem appeared; if we only defined processes, the evolution would be complex. Normally, processes have to be changed or implemented in order to be adapted to the real needs of the organization. The definition based on static processes was not a suitable solution. For this reason, we started working on a MDE definition and created the metamodel defined in [23]. Thus, if we want to change an activity, extend the environment with new processes or even change the orchestration of the process map, we have a proper tool to do it.

5. Project References

In this section, we will analyse the use of NDT-Suite on the recent projects in which we have participated. First of all, we will quickly delimit the scope and heart of each project. This way, the advantages of using NDT-Suite will be easier to understand. Secondly, we will describe the conclusions obtained after the development of each project. Finally, we will state the pros and cons of NDT and our research in order to improve it.

Thanks to NDT has been applied in many projects, companies have issued different feedbacks for each NDT project, which have been very useful for us. These comments are not always positive and some of them criticize the system. Such comments are precisely the most important, because they help us make our tool more efficient for our future projects.

NDT-Suite has and are been used in different companies such as the Andalusian Regional Ministry of Culture; the Andalusian Regional Ministry of Health and Social Welfare; Emasesa and Airbus, among others. Working “hand to hand” with NDT-Suite, all collaborative companies use Enterprise Architect as a tool for managing their requirements and the software/product lifecycle. NDT-Suite provides an additional behavior to Enterprise Architect, defining a particular profile for each project (NDT-Profile), adjusting the tool to the enterprise’s needs and validating the quality of the work done (NDT-Quality), and managing all these aspects with NDT-Driver.

Now, we are going to describe below how we used NDT-Suite in three specific projects: Mosaico [30] [31], AQUA-WS (Aqua-Web Services) [32], and CALIPSOneo [33].

The Andalusian Regional Ministry of Culture started to develop the **Mosaico project** in 2004. The idea of this Web application was born from the need to manage all the information on historic heritage in Andalusia. Before Mosaico, there were several systems in charge of managing this information, what caused a lot of problems since the information was distributed, disconnected and different users worked in different platforms. Consequently, the growing need of managing and maintaining historic heritage promoted a project like this. Mosaico was developed by two important companies and it covered 5670 requirements, out of which 3253 were functional requirements.

In this project, we used the first version of NDT-Quality, but we realized that was not enough to ensure the quality of this enormous project.

At present, this project is allowing us applying a general model-driven approach for the systematic detection of requirements inconsistencies. In addition, this approach is adapted and extended to improve the NDT methodology and NDT-Suite. In fact, results obtained after applying this approach in a project like of Mosacio, open a very attractive line for our research works. As NDT is applied in a high number of companies, our next step is the inclusion of this approach in NDT-Suite, particularly in NDT-Quality and NDT-Merge. This implementation will improve NDT-Quality. We want to test the implementation in a complementary set of real projects to try to measure, in an objective way, the number of consistencies that are detected. This work will be made with companies and analyst groups that use NDT for Web software development.

Moreover, the **AQUA-WS project** was carried out in collaboration with Emasesa in 2011. The former, AQUA-WS stems from the needs of unifying all the systems that operated at Emasesa, a local company that manages the water cycle in Seville (Spain). This company aims to provide and guarantee quality in the water supply, solving each type of incidence that could appear and controlling the proper utilization of water.

Initially, Emasesa handles the following independent software systems:

- AQUA-SiC. Clients’ data management. This system is a local application.
- AQUA-ReD. Water networks management. This system is based on Web technologies.
- AQUA-SigO. Work and project management. This system is based on Web technologies.

The new system, AQUA-WS, also deals with Web technologies (particularly Java J2EE), but its main goal was to design a Decision Support Systems (DSS), to ensure the correct communication between AQUA Core and other subsystems used by Emasesa, providing an easy-to-use FrontEnd interface. **Figure 5** shows the architecture of AQUA-WS. AQUA Core, FrontEnd, DSS and the communication with transversal systems was based on the use of SAP ERP (SAP Enterprise Resource Planning) [34] and SAP BW (SAP Netweaver Business Warehouse) [35] on a GIS (Geographical Information System) and a documentary resources system.

In this project, NDT-Suite aims to automate the steps among each software engineering phase (Requirements, Analysis, Design, Implementation, Testing and Maintenance). The project has a high number of complex Functional Requirements; more than 70% have more than 15 steps, whereas the complete project has 1808 Functional Requirements. To perform tests for each requirement of the project implies 2300 work hours. Time has been re-

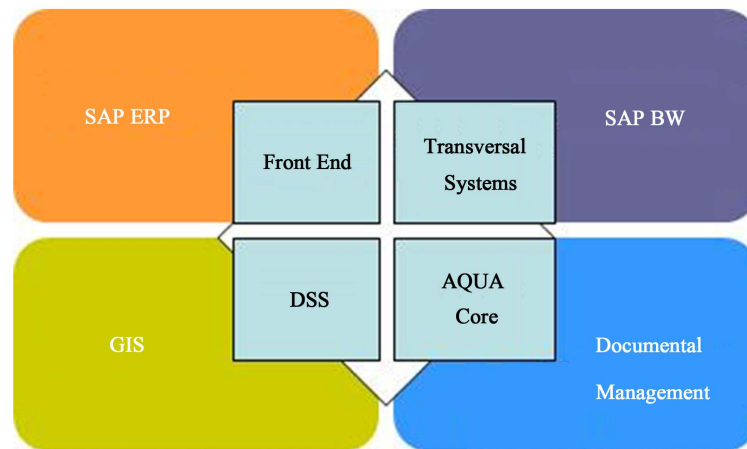


Figure 5. AQUA-WS architecture.

duced to 500 hours by using NDT-Driver to generate the test from the requirements defined. Therefore, NDT-Driver saves 1800 working hours.

The latter, the **CALIPSONEO project**, was developed in 2013 with airbus in Seville (Spain). CALIPSONEO (Advanced Aeronautical Solutions Using PLM Processes & Tools) is an ambitious project that tends to identify, through a comprehensive requirements collection process using PLM (Product Life-Cycle Management) [36], the software existing in the market. That will allow industrial engineers to define, simulate, optimize and validate the aeronautical assembly processes in a 3D virtual environment, before being implemented in a real shop floor.

In order to effectively manage the work required to complete Calipsoneo, it will be subdivided into three individual subprojects: MARS (Automated Shop-Floor Documentation Updating System), PROTEUS (Process Structure Generation and Use) and ELARA (Generalization to Assembly Oriented Authoring Augmented Reality). These subprojects are independent and teams involved in each of them also differ. However, subprojects have to be coordinated and they have to be correctly integrated because they have common actors who demand common functionality.

One of the main characteristics of this project, which makes it very attractive to our team, is the fact to be more centered on requirements and work policies than in code. For this reason, we need to adapt NDT-Suite to the project's requirements. Another characteristic is the heterogeneity of the project team, as they are located in different cities in Spain and they can implement the project with very different skills and experiences.

As mentioned before, CALIPSONEO is composed of 3 subprojects. It is worth mentioning that NDT-Profile was modified to take into account this factor, by extending the original profile to offer a 'project composed of subprojects'. Monthly, a merge task was performed to compile the information of the 3 projects on a master NDT-Profile project.

CALIPSONEO project's MARS and PROTEUS deal with developing a strong work guideline or a PLM methodology, so that, to perform these projects with NDT-Suite, it was very important to lead the whole work to the requirement, analysis and testing phases.

In relation to testing in CALIPSONEO, NDT-Suite was used as a reference, so that a new guideline could be provided to perform tests efficiently and in less time, by working together with the development team as shown in [37]. NDT-Driver was very powerful and helped reduce human errors during the test definition phase.

6. Conclusions and Future Work

The application of MDE becomes complex, monotonous and very expensive, if no software tool automates the process. In consequence, NDT has defined a set of supporting tools called NDT-Suite in order to meet this need. In the last ten years, NDT and NDT-Suite have been used in a large number of real projects. The main advantage of NDT-Suite is that it reduces the cost of ensuring quality and traceability of the deliverables carried out during the project development phases. This objective is achieved by means of NDT-Quality. In addition, we continue improving NDT-Quality. For example, we are studying how this tool may allow defining Key Process

Indicators (KPI) in order to improve the quality assurance process when NDT methodology is used in real projects. Besides, we are also working in the use and implementation of QuEF (Quality Evaluation Framework) [38].

We are examining and developing NDT-Merge, which is closely related to NDT-Quality, in the quality assurance context. NDT-Merge aims to identify possible conflicts and conciliate requirements on large projects where each working team runs tasks in a different module system.

Moreover, NDT-Suite also reduces the cost of deliverables of one phase because they are obtained from other deliverables of the previous phase. This objective is achieved using NDT-Driver, which is a very powerful and useful tool to define information systems. As long as we know that NDT-Driver is not functional enough, we propose several improvements as future research. On the one hand, we are studying how to improve and enrich test generation from Functional Requirements (which are defined using UML Activity Diagrams) with new techniques for selecting and reducing redundant testing paths. On the other hand, we suggest analyzing how to incorporate heuristics in the generation of some models. For instance, in the generation of the Navigation Model of the Analysis Phase, NDT-Driver should identify aspects, such as the graph is not connected, by means of the Warshall algorithm.

Additionally, this paper briefly presents a solution for the business process definition focused on a model-based approach. This solution is implemented in Enterprise Architect and a concrete solution, named NDTQ-Framework, is offered. NDTQ-Framework has been used in several real projects from which some relevant conclusions can be deduced.

Firstly, a model-based mechanism to define business processes can be very useful but, if we do not offer concrete syntaxes to represent them, it is difficult for companies to use it. UML Profiles and UML-based tools seem to be good options to represent business processes in the software processes environment, since the development team usually knows this notation.

However, our NDTQ-Framework has one limitation: today, it focuses on proposing a software process definition mechanism. In this context, orchestration and execution processes are manually and/or unilaterally performed by each involved role in each process. This fact makes business processes monitoring and measurement become a difficult task. For this purpose, we are already doing research on how to suggest a BPM lifecycle for business software in order to perform continual improvement. This BPM lifecycle is supported by PLM₄BS [39] (Product Lifecycle Management for Business-Software): a tools-based and MDE-based framework for software processes modelling, orchestrating, executing and monitoring.

To conclude, we are working on how to extend NDT to provide support at the implementation phase of a Web project. Only few Web Engineering methodologies support the systematic development of Web applications with a mature CASE tool. Thus, some methodologies such as OO-H with Visual-WADE [40], OOWS with Olivanova or UWE with UWE4JSF [41], among others, constitute specific solutions that offer code generation in a MDE context.

Acknowledgements

This research has been supported by MEGUS project (TIN2013-46928-C3-3-R) of the Ministerio de Ciencia e Innovación and NDTQ-Framework project (TIC-5789) of Junta de Andalucía, Spain. This paper has also been supported by the Universia Foundation through its student grant for PhD students.

References

- [1] Schmidt, D.C. (2006) Model-Driven Engineering. *IEEE Computer, Computer Society*, **39**, 25-31. <http://dx.doi.org/10.1109/MC.2006.58>
- [2] Escalona, M.J. and Aragón, G. (2008) NDT. A Model-Driven Approach for Web Requirements. *IEEE Transactions on Software Engineering*, **34**, 377-390. <http://dx.doi.org/10.1109/TSE.2008.27>
- [3] OMG (2008) Documents Associated with Meta Object Facility (MOF) 2.0 Query/View/Transformation. Object Management Group, Needham. <http://www.omg.org/spec/QVT/1.0/>
- [4] ATL (2014) ATL—A Model Transformation Technology. <http://www.eclipse.org/atl/>
- [5] NDT-Suite (2014) www.iwt2.org
- [6] Domínguez-Mayo, F.J., Escalona, M.J., Mejías, M., Ross, M. and Staples, G. (2014) Towards a Homogeneous Characterization of the Model-Driven Web Development Methodologies. *Journal of Web Engineering*, **13**, 129-159.

- [7] Domínguez-Mayo, F.J., Escalona, M.J., Mejías, M. Ross, M. and Staples, G. (2012) Quality Evaluation for Model-Driven Web Engineering Methodologies. *Information Software Technology*, **54**, 1265-1282. <http://dx.doi.org/10.1016/j.infsof.2012.06.007>
- [8] Rossi, G. and Schwabe, D. (2008) Modelling and Implementing Web Applications with OOHD. Web Engineering: Modelling and Implementing Web Applications. Web Engineering: Modelling and Implementing Web Applications (Human-Computer Interaction Series), 109-155. http://dx.doi.org/10.1007/978-1-84628-923-1_6
- [9] Koch, N., Knapp, A., Zhang, G., and Baumeister, H. (2008) UML-Based Web Engineering. In: Rossi, G., Pastor, O., Schwabe, D. and Olsina, L., Eds., *Web Engineering: Modelling and Implementing Web Applications*, Springer, Berlin, 157-191.
- [10] UML (2005) Modeling Language: Superstructure. Specification, OMG, Needham. <http://www.omg.org/cgi-bin/doc?formal/05-07>
- [11] ArgoUWE (2014) ArgoUWE—CASE Tool for Modelling Web Applications. <http://uwe.pst.ifi.lmu.de/toolargoUWE.html>
- [12] MagicUWE (2014) MagicUWE—UWE Plugin for MagicDraw. <http://uwe.pst.ifi.lmu.de/toolMagicUWE.html>
- [13] Ceri, S., Fraternali, P. and Bongio, A. (2000) Web Modelling Language (WebML): A Modelling Language for Designing Web Sites. *Computer Networks*, **33**, 137-157.
- [14] OMG (2011) BPMN, Business Process Modelling Notation, Version 2.0. Object Management Group. <http://www.omg.org/spec/BPMN/2.0/>
- [15] WebRatio (2014) <http://www.webratio.com/portal/content/es/home>
- [16] Fons, J., Pelechano, V., Albert, M. and Pastor, O. (2003) Development of Web Applications from Web Enhanced Conceptual Schemas. *Lecture Notes in Computer Science*, **2813**, 232-245.
- [17] ISO/IEC (2012) ISO/IEC 19507:2012 Information Technology—Object Management Group Object Constraint Language (OCL). International Organization for Standardization, Geneva.
- [18] Enterprise Architect (2014) UML Modeling and Lifecycle Tool Suite. www.sparxsystems.com
- [19] García-García, J.A., Escalona, M. J., Ravel, E., Rossi G. and Urbieto, M. (2012) NDT-Merge: A Future Tool for Conciliating Software Requirements in MDE Environments. iiWAS, Bali, 177-186.
- [20] García-García, J. A., Escalona, M. J., Cutilla, C.R. and Alba, M. (2011) NDT-Glossary. *Proceedings of the 13th International Conference on Enterprise Information System (ICEIS 2011)*, No. 13, 170-175.
- [21] Armario, J., Gutiérrez, J., Alba, M., García-García, J.A., Vitorio, J. and Escalona, M.J. (2012) Project Estimation with NDT. In: Hammoudi, S., van Sinderen, M. and Cordeiro, J., Eds., *ICSOF-International Conference on Software Paradigm Trends*, Rome, 24-27 July 2012, 120-126.
- [22] OMG (2011) Meta Object Facility (MOF™) Core. Object Management Group. <http://www.omg.org/spec/MOF/>
- [23] Ponce, J., García-Borgoñon, L., García-García, J.A., Escalona, M.J., Domínguez-Mayo, F.J., Alba, M. and Aragon, G. (2013) A Model-Driven Approach for Business Process Management. *Covenant Journal of Informatics and Communication Technology (CJICT)*, **1**, 32-52.
- [24] ISO/IEC (2007) ISO/IEC TR 24744:2007 Software and Systems Engineering Lifecycle Management Guidelines for Process Description. International Organization for Standardization, Geneva.
- [25] Jong, A. and Kolthof, A. (2008) Fundamentos de ITIL, Vol. 3. Van Haren Publishing, Zaltbommel.
- [26] Bosch, J. (2009) From Software Product Lines to Software Ecosystems. *SPLC' 09 Proceedings of the 13th International Software Product Line Conference*, San Francisco, 24-28 August 2009, 111-119.
- [27] ISO/IEC (2014) ISO/IEC 29119 Software Testing. The New International Software Testing Standard. www.softwaretestingstandard.org
- [28] ISO/IEC (2008) ISO/IEC 9001:2008 Quality Management Systems—Requirements. International Organization for Standardization, Geneva.
- [29] PMI (2014) Project Management Body of Knowledge. www.pmi.org/PMBOK-Guide-and-Standards.aspx
- [30] ISO/IEC (2013) ISO/IEC 27001—Information Security Management. International Organization for Standardization, Geneva.
- [31] Escalona, M.J., Urbieto, M., Rossi, G., Garcia-Garcia, J.A. and Luna, E.R. (2013) Detecting Web Requirements Conflicts and Inconsistencies under a Model-Based Perspective. *Journal of Systems and Software*, **86**, 3024-3038. <http://dx.doi.org/10.1016/j.jss.2013.05.045>
- [32] Cutilla, C.R., García-García, J.A., Alba, M., Escalona, M.J., Ponce, J. and Rodríguez, L. (2011) Aplicación del paradigma MDE para la generación de pruebas funcionales; Experiencia dentro del proyecto AQUA-WS. 6ª Conferència Ibèrica

de Sistemas e Tecnologias de Informação ICSoft-International Conference on Software Paradigm Trends, Chaves, 15-18 June 2011, 811-815.

- [33] Escalona, M.J., Garcia-Garcia, J.A., Mas, F., Oliva, M. and Del Valle, C. (2013) Applying model-Driven paradigm: CALIPSOneo Experience. *Proceedings of the Industrial Track of the Conference on Advanced Information Systems Engineering 2013 (CAiSE'13)*, Vol. 1017, Valencia, 21 June 2013, 25-32.
- [34] SAP ERP (2014) www.sap.com/ERP
- [35] SAP BW (2014) <http://www.sap.com/solutions/bench mark/bw.epx>
- [36] Stark, J. (2011) *Product Lifecycle Management: 21st Century Paradigm for Product Realisation*. Springer Science, Berlin.
- [37] Salido, A., García-García, J.A., Ponce, J., Gutiérrez, J.J. and Oliva, M. (2013) Gestión de pruebas en CALIPSOneo: Una solución MDE. V Congreso Internacional de Computación y Telecomunicaciones, Lima.
- [38] Domínguez-Mayo, F.J., Escalona, M.J., Mejías, M., Ross, M. and Staples, G. (2012) A Quality Management Based on the Quality Model Life Cycle. *Computer Standards & Interfaces*, **34**, 396-412.
<http://dx.doi.org/10.1016/j.csi.2012.01.004>
- [39] García-García, J.A., Escalona, M.J., Mejías, M. and García-Borgoñón, L. (2013) Model-Based Approach for Software Processes Modelling in PLM4BS. *Enterprise Information Systems Journal*, Under review.
- [40] Visual-WADE (2014) <http://visual-wade.software.informer.com/>
- [41] UWE4JSF (2014) <http://uwe.pst.ifi.lmu.de/toolUWE4JSF.html>