Covenant Journal of Informatics and Communication Technology (CJICT) Vol. 1, No. 2, December, 2013.

A Model-Driven Approach for Business Process Management

J. Ponce[1],
L. García-Borgoñon[1,2],
J. García-García[1],
M.J. Escalona[1],
F.J. Dominguez-Mayo[1],
M. Alba1,
G. Aragon[1]

[1] Web Engineering and Early Testing Group.
University of Seville. Seville, Spain
[2] Insituto Tecnologico de Aragon. Zaragoza, Spain.

Contact(s):. josepg@us.es, laurag@ita.es, julian.garcia@iwt2.org, mjescalona@us.es, fjdominguez@us.es, manuel.alba@iwt2.org, gustavo.aragon@iwt2.org

Abstract: The Business Process Management is a common mechanism recommended by a high number of standards for the management of companies and organizations. In software companies this practice is every day more accepted and companies have to assume it, if they want to be competitive. However, the effective definition of these processes and mainly their maintenance and execution are not always easy tasks. This paper presents an approach based on the Model-Driven paradigm for Business Process Management in software companies. This solution offers a suitable mechanism that was implemented successfully in different companies with a tool case named NDTQ-Framework.

Keywords/Index Terms**:** Model-Driven Web Engineering, Web Engineering, Web Development Methodologies, Business Process Management

1. Introduction

Nowadays, the Business Process definition and management constitute a more used mechanism in enterprise management as well as in software companies. Besides, in software companies, it is used for the application of a high number of quality or security assurance standards references, among others.

Business Process Management (BPM) is a well-established research and practice field, which intends to provide assistance for operational business processes from the Design to the Enactment phases (Pillain et al. 2011). Different techniques for business process modeling and business process execution as well as for their relationship have been proposed. Business Process Modeling Notation (BPMN) (OMG 2011) has become a widely used standard in process modeling and

Business Process Execution Language (BPEL) (OASIS 2013) has consolidated as the language for business process execution.

However, the BPM in software development context is not as simple as it appears to be. Software development is constantly evolving and it usually incorporates new life cycles, technologies or development teams, among other aspects. Software processes consist of several iterations or versions of products and use different development tools. They are highly influenced by a large number of factors becoming more complex and unpredictable than traditional production processes.

In the Software Process Engineering area, the main objective focuses on improving software development practice by proposing: a) better ways for designing the developer organization processes and b) better ways for improving this organization both individually and as a whole (Acuña and Juristo, 2005). For this purpose, software process research is divided in two lines: software process modelling and software process evaluation and improvement. The latter includes all efforts from the software community related to maturity models and standards like ISO 9001:2008 (ISO 2008), Capability Maturity Model Integration (CMMI) models (Chrissis et al.2003) and ISO/IEC 15504:2004 (ISO 2004). In recent years, it has become a very extensive and popular field and therefore there is a lot of related work in this area.

The finding of techniques, mechanisms and mainly tools to make easier the definition, maintenance and improvement in this environment is a research area that is being demanded by the enterprise since current solutions are still too manual.

This paper presents an approach, based on the Model-Driven paradigm, implemented in a tool case named NDTQ-Framework. This approach uses the concepts of metamodels to define software processes as well as it uses this resource for the integration, orchestration and improvement of software processes. NDTQ-Framework integrates some of the most relevant maturity models and standards for software development and it is prepared for being extended with new processes or business areas.

The paper is structured as follows. In Section 2, related work is presented and in Section 3 the background of our approach is introduced. The solution, NDTQ-Framework, is presented in detail in Section 4 and in Section 5 some enterprise references are numbered. The paper finishes with conclusions and future work.

2. Related Work

In order to propose the basis of this approach, we have to analyze which are the standards to be followed by software companies or which are the most relevant standards in this context. This section presents models, standards or references to be considered.

2.1. ISO Related Standards

ISO 9001:2008 specifies requirements for a quality management system where an organization needs to demonstrate its ability to consistently provide products that meet customer and applicable statutory and regulatory requirements. It also aims to enhance customer satisfaction through the effective application of the system, including processes for continual improvement as well as customer conformity assurance and applicable statutory and regulatory requirements. All ISO 9001:2008 requirements are generic and they intend to be applicable to all organizations, regardless of type, size and product provided. Where any requirement(s) of ISO 9001:2008 cannot be applied due to the nature of an organization and its product, this can be considered for exclusion. In cases where exclusions are made, claims of conformity to ISO 9001:2008 are not acceptable, unless these exclusions are limited to requirements within Clause 7 and they do not affect the organization's ability or responsibility to provide products that meet customer and applicable statutory and regulatory requirements.

ISO/IEC 12207:2008 (ISO 2013) establishes a common framework for software life cycle processes, with well-defined terminology, that can be referenced by the software industry. It contains processes, activities, and tasks to be applied during the acquisition of a software product or service and during the supply, development, operation, maintenance and disposal of software products. Software includes the software portion of firmware. This standard applies to the acquisition of systems, software products and services, the supply, development, operation, maintenance, and disposal of software products and the software portion of a system, either internally or externally performed in an organization. Those aspects of system definition needed to provide the context for software products and services are included. ISO/IEC 12207:2008 also offers a process that can be executed for defining, controlling and improving software life cycle processes. The processes, activities and tasks of ISO/IEC 12207:2008 - either alone or in conjunction with ISO/IEC 15288:2008 (ISO 2013) - may also be applied during the acquisition of a system that contains software.

An increasing number of international, national and industry/ industrial standards describe process models (Prikladnicki et al. 2006)). These models are developed for a range of purposes including process implementation and assessment. The terms and descriptions used in such models vary in format, content and level of prescription. ISO/IEC TR 24774:2010(ISO 2013) presents guidelines for the most frequently used elements to describe a process: title, purpose, outcomes, activities, tasks and information items. Whilst the primary purpose of ISO/IEC TR 24774:2010 is to encourage consistency in standard process reference models, its rules can be applied to any process model developed for any purpose.

The new ISO/IEC 29119 Software Testing (still unpublished) (ISO 2013) is another ISO standard to consider. The aim of this new standard is to provide one definitive standard that captures vocabulary, processes, documentation and techniques for the entire software testing life cycle. This standard will support testing on any software development or maintenance project, from organizational test strategies and test policies, project and test strategies phase and plans related to test case analysis, design, execution, reporting and beyond.

2.2. Other Related Standards

The CMMI is a process improvement approach that provides organizations with the essential elements of effective processes, which will better their performance. CMMI-based process improvement identifies your organization's process strengths and weaknesses and turns weaknesses into strengths. CMMI applies to teams, work groups, projects, divisions and entire organizations. Find out why CMMI has been adopted by so many organizations worldwide. The published CMMI appraisal results searchable database of appraisal results lists hundreds of organizations that use CMMI. CMMI models are collections of best practices that help organizations to dramatically improve effectiveness, efficiency and quality. These products, or CMMI solutions, consist in practices which cover topics that include causal analysis; configuration management; quality assurance; verification and validation; risk management; requirements management; supplier management; project management; interface compatibility; make, buy, or reuse analysis; capacity management; availability management; disaster recovery, data collection, process performance and some other topics. View the inclusive set of CMMI process areas to get a more complete picture of the topics it covers.

ITIL (Information Technology Infrastructure Library) (ITIL 2013) is a widely accepted approach and provides a cohesive set of best practice for the IT service management. Information Technology Service

Management (ITSM) (ITSM 2013) derives enormous benefits from a best practice approach. Because ITSM is driven both by technology and the huge range of organizational environments in which it operates, it is always evolving. Best practice based on experts advice and input from ITIL users is both current and practical, combining the latest thinking with sound, common sense guidance.

As far as project management practices are concerned, Project Management Body of Knowledge PMBOK (PMBOK 2013) is a standard guide for project management. It represents well-recognized good practices in the professional field while reflecting the continually evolving knowledge in project management.

The Guide to the Software Engineering Body of Knowledge (SWEBOK 2013) has the purpose of provide a validated characterization of the bounds of the software engineering discipline. The Body of Knowledge is subdivided into ten software engineering Knowledge Areas (KA), which are: software requirements, software design, software construction, software testing, software maintenance, software configuration management, software engineering management, software engineering process, software engineering tools and methods and software quality. One of its objectives is to clarify the place and set the boundary of software engineering with respect to other disciplines, and it summarizes the content and relationships of each one of their KAs.

The ISO/IEC 15504:2004, also known as SPICE (Software Process Improvement and Capability dEtermination) (SPICE 2013) is a set of technical standards documents for assess software business processes. It is not only focused on software development processes, but also covers all related processes in six software business areas: organizational, management, engineering, acquisition supply, support and operations. As CMMI with its appraisal part, ISO/IEC 15504:2004 guides the assessment of organizational processes and allows determine the capability level of them. It only defines the assessment process and establish others standards like ISO/IEC 12207:2008 and ISO/IEC 15288:2008 as reference models.

In the field of IT Governance, COBIT (Control Objectives for Information and related Technology) (COBIT 2007) is an important reference framework. Its mission is “to research, develop, publish and promote an authoritative, up-to-date international set of generally accepted information technology control objectives for day-to-day use by business managers, IT professionals and assurance professionals”. The business orientation of COBIT consists of linking business goals to IT goals, providing metrics to measure their achievement and identifying the associated responsibilities of business and IT process owners. It would be positioned at a high level and could coexist with others models.

2.3. Other Reference Models

Pardo et al. 2011 present a systematic review of initiatives and projects published and carried out on the harmonization of multi-model environments such as ISO 9001:2008 (for Software Quality Management), ISO/IEC 90003:2004 (ISO 2013) (to improve the capacity of models), CMMI and ISO/IEC 12207:2008 (for the IT Governance), ITIL, PMBOK and COBIT, Information Security Management Systems such as ISO/IEC 27000:2009 (ISO 2013) and bodies of knowledge such as SWEBOK, amongst others. They also discuss the significant issues regarding this area of knowledge, providing an up-to-date state of the art and identifying possible related research streams. The authors, Pardo et al. 2011 wrote another interesting paper where a tool called HProcessTOOL is presented to guide companies in the harmonization projects by supporting specific techniques and their management by means of controlling and monitoring the resulting harmonization projects. The tool is applied in two case studies and allows the work products, effort, time and roles involved in the harmonization projects, as well as the knowledge generated, to be correctly managed.

In the paper mentioned above, R. Prikladnicki et al. 2006, introduce a reference model for global software development, based on the results found in a case study conducted in two software development units from multinational organizations located in Brazil. A preliminary description of this model was originally published in 2004. The reference model is presented in detail, improving the description and discussing the factors that enable multinationals corporations to operate successfully across geographic and cultural boundaries. At the end, the authors discussed a preliminary evaluation of the reference model usage.

3. Background

Applying the methodologies and related work in software development helps ensure the quality of the results. Frameworks and Metrics v3 (MAP 2013) or modeling languages, such as standard UML (Unified Model Language) (OMG 2013), are widespread in the software development field, and business developers, as customers, apply them in their projects requirements.

However, when analyzing the reality of the developments, there are many problems that, in theory, should not occur. In many cases, the documentation process or the exact application of methodological phases imply a mere formality. In many cases, projects starts being framed in a suitable methodological framework, just delayed, suffering or patched code changes that cause the end, the consistency between the documentation and the system is very low.

This reality has fostered the development of good practice guidelines, standards or work systems to enhance aspects such as quality assurance, traceability or, even, strengthening the testing phase as a testing tool. ISO, CMMI and ITIL are included among them. However, if we put in place a development team who is required to work under a specific methodological environment, according to these quality standards and best practices, by promoting testing phases and lines of communication with users, we realize that the situation seems complex since there are too many fronts to be addressed in a cost effective manner.

This need to combine processes and provide adequate channels, by considering the economic costs of the project time, has meant that in recent years, the Model-Driven paradigm has offered a very attractive solution to solve the problem. When developing Web-oriented, Model-Driven Engineering has resulted in methodologies such as UML Web Engineering (UWE) (Koch et al. 2008), WebML (Web Modelling Language) (Ceri et al. 2009) or OOHDM (Object Oriented Hypermedia Design Methodology) (Rossi et al.2008). They are beginning to offer a good framework for companies. In this context, we have developed a methodology named NDT (Navigational Development Techniques)(Escalona and Aragon 2008) which is a methodological approach based on the Model-Driven paradigm that began working on the Requirements Engineering field, but nowadays covers the entire life cycle. NDT has also been proposed and it is highly accepted in the business environment.

Indeed, we have learned from our experience that offering the development methodology exclusively is not enough. When a company has to face up a project, it has to cope with other aspects apart from the development process. For this reason, we started working on NDTQ-Framework, which covers the definition of five process groups (development processes, maintenance processes, software testing process, project management processes and software quality processes). To define each of these processes, it is necessary to look at different reference standards, as it is presented in the next section.

For this aim, a metamodel based on the ISO/IEC TR 24774:2010 was defined and implemented and later some instances of this metamodel were used to define processes. It aims to define a set of orchestrated and defined processes to provide the working team with all the elements needed to develop their projects under the NDT paradigm. As shown below, each process is described in detail with their inputs and outputs, indicators and metrics to measure progress and quality degrees according to the structure marked by the metamodel.

4. NDT (Navigational Development Techniques)

Initially, NDT was a methodology focused on the Requirements and Analysis phases with particular emphasis on boosting the requirements, their traceability into the analysis, and the transformations execution, which would support the development stage. Hence, NDT dealt with the definition of a set of formal metamodels for the Requirements and Analysis phases. In addition, NDT defined a set of derivation rules that generated the analysis models from requirements models.

NDT evolved as a theoretical proposal. Their metamodels, transformations and tools became the standard MOF and its Query/View/Transformation facility (QVT) (QVT 2013). In this way, NDT was able to interact very efficiently with other methodologies UWE or WebML.

Nowadays, NDT defines metamodels for every phase of the software development life cycle: the Feasibility study phase, the Requirements phase, the Analysis phase, the Design phase, the Implementation phase, the Testing phase, and finally, the Maintenance phase. Besides, it states new transformation rules to systematically generate models.

However, it was impossible to use this framework in the enterprise environment. After carrying out various surveys among some companies' members of the staff, it was noticed that in companies, the use of metamodels, transformations and other elements is not real and this technology seems too abstract for them. We conclude that a methodology will succeed in enterprise environments, if it provides software tools that facilitate its use and ensure the quality of the results. In this sense, NDT methodology provides support to each of the phases of the life cycle of software development through a set of software tools. For this reason, in 2004 we started working on a new set of tools to support these NDT aspects in two lines: offering both an enterprise solution for NDT usage and a way to validate transformations rules and metamodels.

After considering different possibilities, some UML profiles were developed for each metamodel of NDT. These UML profiles were defined in a UML-based tool named Enterprise Architect1 and then, the

first tool for NDT-Suite (NDT-Suite 2013), NDT-Profile, was developed.
To select Enterprise Architect was not easy. In fact, our research team carried out a comparative study in liaison with the Junta de Andalucía (Andalusian Regional Government), which concluded that it was the tool that offered the best value for money.

Therefore, UML profiles constituted our best choice since some empirical studies concluded that it was the easiest mechanism for people working in companies. UML is commonly used in software companies as the best option, since development teams know its notation due to the fact that they usually work with UML-based tools.

Nevertheless, Enterprise Architect involves a problem when defining transformations. Enterprise Architect or other UML-based tools analyzed do not support QVT transformations. Thus, NDT transformations were translated into Java and implemented in a tool that was included in NDT-Suite, named NDT-Driver (Garcia-Garcia et al. 2013). This solution, NDT-Profile and NDT-Driver, provides development teams with a tool environment to define models as well as an engine to execute transformations. These seeds prepare the NDT environment both for research and practical usage.
Today, NDT-Suite comprises the following free Java tools:

- NDT-Profile is a specific profile for NDT developed by means of Enterprise Architect. NDT-Profile offers the chance of gathering all the elements defining NDT easy and quickly, as they are integrated within the Enterprise Architect tool.
- NDT-Quality is a tool that automates most of the methodological review of a project developed with NDT-Profile. It checks both, the quality of NDT methodology in each phase of software life cycle and the quality of traceability of the MDE rules of NDT.
- NDT-Driver implements all the automatic procedures for carrying out each of the QVT transformations defined in NDT.
- NDT-Prototype is a tool designed to automatically generate a set of XHTML prototypes from the navigation models of a project described in the Analysis phase, developed with NDT-Profile.
- NDT-Glossary (Garcia-Garcia et al. 2011) implements an automated procedure that generates the

first instance of the glossary of terms in a project developed by means of NDT-Profile.

- NDT-Report is a tool to generate PDF and RTF documents from NDT-Profile.

Although applying methodologies, specifically NDT, helps ensure the quality of the results, the real life uncovers many problems that should not take place. Methodologies like Rational Unified Process (RUP) (Krutchen 2007) or standard modelling languages like UML, are very widespread within the scope of software development. Both providers and customers apply and demand them in their projects.

However, in many cases the elaboration of documents or the exact application of methodological phases ends up as mere formality. Sometimes, projects framed in a methodology and NDT context experience delays, changes or patched code. This causes inconsistency between the documentation and the final system.

Despite we have learned several lessons with the real application of NDT (Escalona et al. 2007) two conclusions are especially important for creating the NDTQ-Framework:

1. MDE is a good solution for software development and helps reduce cost and time and increase the final quality and traceability of the software. However, metamodels, transformations and this context are not friendly for development teams, thus in case of being applied, suitable tools have to be developed.
2. The development process is not enough for 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 this reason, we propose our framework, NDTQ-Framework that, based on the development process of NDT, enables to define, manage and improve other processes in software development.

5. NDTQ-Framework

As it was introduced in the previous section, NDTQ-Framework does not only support development processes, but it is an environment for the definition, maintenance and improvement of software processes.

The first step consists in planning how we are going to define each process. For this aim, a metamodel for Software Process Support is proposed. Therefore, we define a MOF metamodel that also incorporates required attributes in ISO/IEC TR 24774:2010 standard. It is issued as a guideline for the process description.

Figure 1 shows the proposed metamodel for Software

Process Support. The 'Process' metaclass is the main class in the metamodel. It represents a set of ordered actions executed by someone in order to produce something. This metaclass needs to incorporate the process title and purpose to comply with ISO/IEC TR 24774:2010.

The 'Activity' metaclass represents the set of ordered actions. An activity can include several activities. This recursive relation is explicitly required at the standard. There are two types of stakeholders involved in performing an activity, who are classified depending on their involvement degree: the 'executes' metaclass represents an actor who carries out the activity while the 'participes' metaclass represents the set of stakeholders who participates, but is not directly responsible of it.

The 'Product' metaclass represents the product obtained as a result when executing a process. This product can be fully developed during the execution of the activities or can be provided from a previous product. It will be defined as an entry and modified within the activity in order to obtain the outcome.

Finally, the 'Metric' metaclass deals with process information elements. Metrics in processes are established with a limited value and it is represented as an instance of 'Indicator'.
This metamodel offers a suitable mechanism for process definition by covering both, the main software process concepts and the ISO/IEC TR 24774:2010 compliant, as well as provides simplicity in order to develop a whole model-based solution.

Following the structure of NDT, an UML profile was defined and implemented in Enterprise Architect, enriching NDT-Profile, to support this metamodel. Thus, we can add a new process in the original approach of NDT.After this initial definition, building NDTQ-Framework requires defining its structure, as it is presented in the next section.

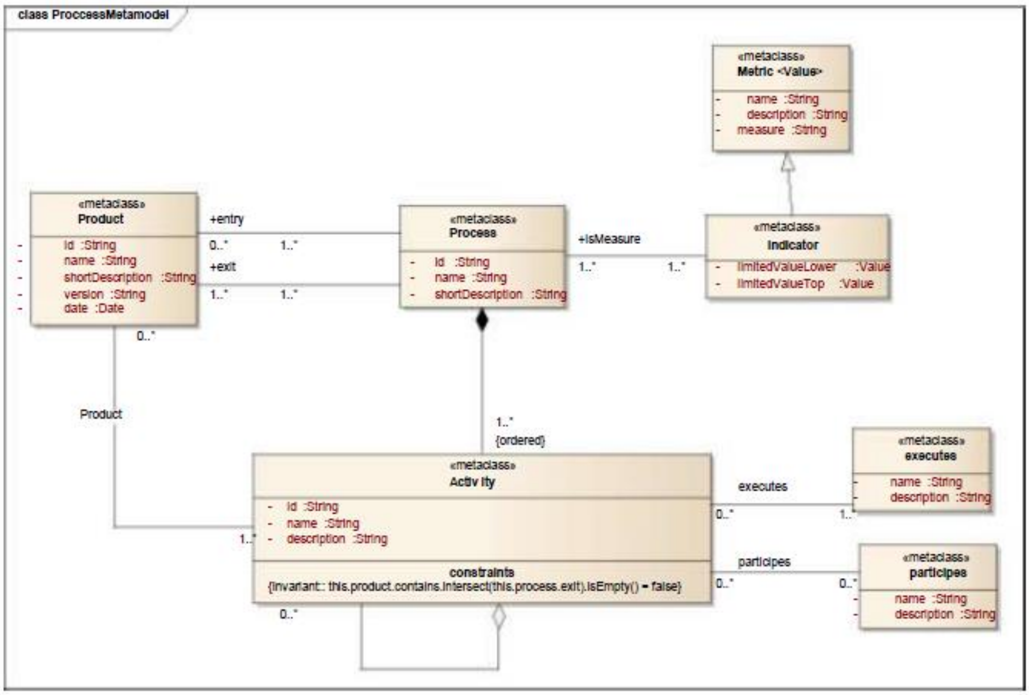

FIGURE 1. PROCESS METAMODEL

5.1. Framework Structure

The following subsections summarize other processes involved in the framework, as well as their stakeholders, that is, the actors. We will also study the deliverables or end products resulting from the use of the framework.

5.1.1. Process Map

The term Process Map is extracted from ISO 9001:2008. It represents an overview of all processes included in the framework and shows the relationships established between them in a generic way. Figure 2 shows NDTQ-Framework Process Map. In the framework, however, has decided to wean two of these activities in the development process: the Testing and Maintenance phases. Thus, the development process encompasses the following processes:

a. Viability Study
b. Requirements Engineering
c. System Analysis
d. System Design
e. Construction and implementation.

It is worth mentioning that NDT implements two ideas, applied in NDTQ-Framework, which are supported by the results obtained. The first one deals with the separation between the Requirements Engineering and Analysis phases. This idea is promoted and justified by a subsequent process is the monitoring of customer satisfaction. Since the end user is actively involved in Requirements Engineering, and due to the fact that NDT gives paramount importance to this phase, following the Requirements Engineering tracks customer satisfaction.

Another relevant aspect is that construction and implementation are performed in a single phase. This is because, as seen just below, NDTQ-Framework supports the development of life cycles. Keeping these phases together will facilitate adaptation to life cycle, incremental, iterative or agile methodologies based.

Each and every one of these processes will be detailed in Section 5.

5.1.2. Stakeholders

Following ISO / IEC TR 24774:2010 and ISO / IEC 12207:2008 guidelines is necessary to indicate which actors are involved in which tasks in the framework. For this aim, NDTQ-Framework includes a diagram describing all the roles and participants who collaborate and act in the process presented in Figure 3.

Actors are structured in two groups in NDTQ-Framework: the software project team and the quality assurance team.

5.1.3. Products

NDTQ-Framework considers three types of deliverables:

- Deliverables own NDT and are generated with the tools of the methodology: document system requirements, analysis, viability, design, test plan, incidence, etc.
- Documentary deliverables or processes are generated and can be represented in different formats, such as project plan, system architecture document, satisfaction survey, implementation manual, training plan, review reports and other relations with the documentary section.

Deliverables generated code especially in the construction phase. They are software products as the prototype, the unit test integration, databases and source code.

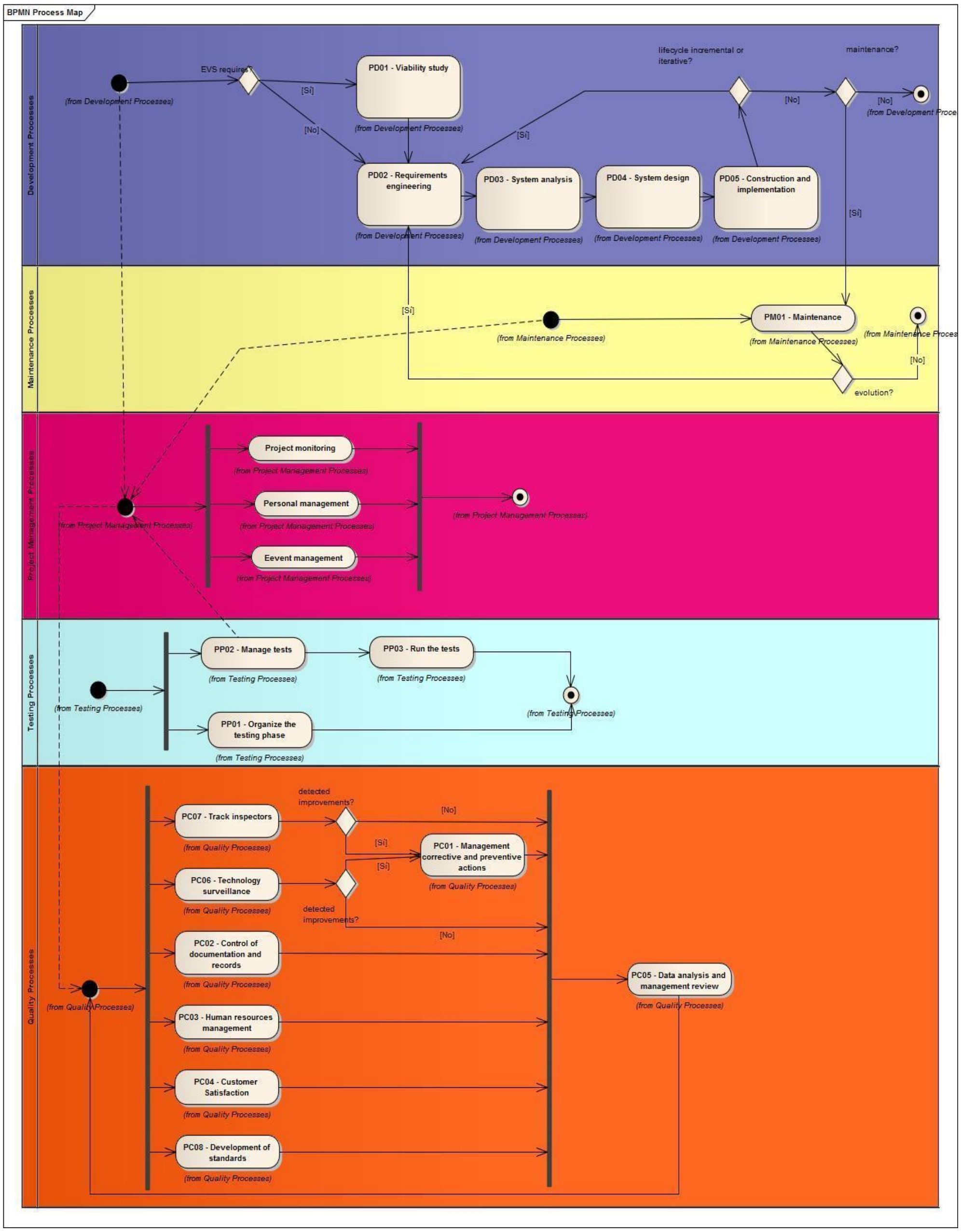
BPMN Process Map
Development Processes
EVS requires?
[Si]
[No]
PD01 - Viability study
(from Development Processes)
PD02 - Requirements engineering
PD03 - System analysis
PD04 - System design
PD05 - Construction and implementation
lifecycle incremental or iterative?
maintenance?
[No]
[Si]
Maintenance Processes
PM01 - Maintenance
(from Maintenance Processes)
evolution?
Project Management Processes
Project monitoring
Personal management
Eevent management
(from Project Management Processes)
Testing Processes
PP02 - Manage tests
PP03 - Run the tests
PP01 - Organize the testing phase
(from Testing Processes)
Quality Processes
PC07 - Track inspectors
PC06 - Technology surveillance
PC02 - Control of documentation and records
PC03 - Human resources management
PC04 - Customer Satisfaction
PC08 - Development of standards
detected improvements?
PC01 - Management corrective and preventive actions
PC05 - Data analysis and management review
(from Quality Processes)

FIGURE 2. NDTQ-FRAMEWORK PROCESS MAP

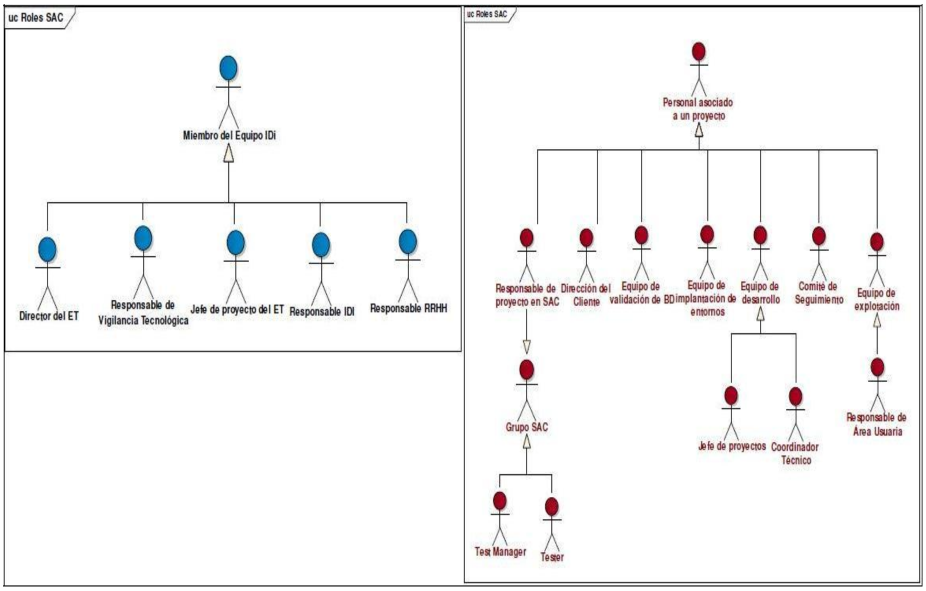

FIGURE 3. ROLES AND PARTICIPANTS IN NDTQ-FRAMEWORK

5.1.4. Group of processes

Each process supported in NDTQ-Framework is defined as instance of the metamodel presented in Figure 1. For every process, NDTQ-Framework defines a set of metrics which measures the effectiveness of the application in the process, the collection of process inputs, the collection of process outputs and the list of supporting tools, as well as provides an activity diagram defining all the tasks and roles involved in the process. NDT-Framework also offers a tool, called NDT-Quality, to ensure the quality of the artifacts generated in each process. There are five groups of processes in NDT. These groups are presented in next sections.

5.1.4.1. Development processes

Development processes group is the backbone of NDT-Framework. All other groups of processes support this group. The development process is composed of five processes. The relations among processes do not entail a waterfall life cycle, whereas the development processes from NDT-Framework have been applied successfully with several life cycles. The five processes from the development group are described in next paragraphs.

- Study of Viability. The aim of this process is to determine the convenience of initiating a new project. This process is not mandatory in the NDT-Framework, but it is very useful when the new project introduces either new technology or a new environment.

- Requirement Engineering. This process intends to develop the requirements artifacts defined by NDT. This is the most important process for all other processes are based on the artifacts generated in this process.
- Analysis of the System. In this process and in the following process too, a first version of artifacts is generated automatically from the results obtained in the previous process using NDT-Suite. These artifacts are then, reviewed, erased, modified or detailed.
- Design of the System. The aim of this process is to define a set of design artifacts, like architecture, system design and user interfaces, from the artifacts generated after the analysis process. This process is applied to every technological stack. However NDT-Suite (the suite of tools supporting NDTQ-Framework) only supports Java and Model-View Controller architectural pattern.
- Construction and implementation. The process intends to configure the development, pre-production and production environments and manage the codification and deployment of the system.

5.1.4.2. Quality assurance processes

Quality is one of the main goals in NDTQ-Framework. The quality process in NDT has been set according to ISO 9001:2008 and UNE 166002:2006 (UNE 2013). The Framework incorporates eight quality processes described as follows:

- Management of Corrective and Preventive Actions. It controls all improvement actions proposed either by the team or other participants.
- Documentation Management. It manages a Content Management System (or other similar platform) and the protocols for storing, retrieving and sharing the relevant information for the process. Relevant information means technical procedures, templates, and other process-supporting material.

- Human Resources Management. It manages the teams assigned to the projects. Fourth process is Customer Satisfaction which measures the satisfaction of all the roles involved in the process of NDTQ-Framework, not only customer, despite its name, using queues and forms.
- Quality monitoring. This process deals with measuring customer satisfaction. NDTQ-Framework must establish different ways to measure the satisfaction of all the roles involved in the processes within the framework.
- Data Analysis and Direction Review. It defines the protocols to perform these two tasks once a year, as it is recommended in ISO 9001:2008.
- Technological Vigilance. It promotes technological improvements among the members of the research group. For this reason, the research groups meet once every two months to share experiences about working on projects and suggesting improvements from solving the problems they have found.
- Inspector Control. It measures and controls the metrics of all the other processes involved in NDT-Framework.
- Guideline Creations. It allows improving NDT-Framework with new documentation and ideas.

5.1.4.3. Management processes

Management processes in NDT-Framework are based on the guidelines and processes defined in PMBOK. There are 3 management processes, which are described in next paragraphs:

- Process Control. This is the most complex process in the organization. Its main goal is to synchronize all activities performed in a project, manage its life cycle and synchronize all resources.
- Event Management. It focuses on controlling and spreading to the right target any event appearing during the development.
- Project Team Management. This process controls the incidents and needs happened to the people conforming the project team.

5.1.4.4. Test processes

Test process in NDTQ-Framework is based on the early testing research performed by the IWT2 group (the same group that created and manage NDTQ-Framework) and the standard ISO/IEC 29119. The test processes group is composed of the three following processes:

- Test Case Definition. It defines the test cases management policy in a project. This goal also includes identifying the testing tool to support testing tasks during the project.
- Test Case Management. It restricts the scope and level of detail to execute the testing phase from the development process.
- Test Case Execution. It delimits the activities, according to test policies and test plan, for executing test cases.

5.1.4.5. Maintenance Processes

The maintenance of information systems is driven in production by only one process in NDT-Framework. This process defines all inputs, outputs, metrics and tasks needed to drive and control bugs improvements and corrections. This process is composed of eight activities involving the development of a service level agreement, by defining the scope of the change and developing a plan, which includes the execution of test cases in order to assure that the change is successfully done and no new bugs have been introduced in the system

6. Enterprise Experience

In the last ten years, NDT and NDT-Suite have acted in a high number of real projects in Spain. In fact, they are currently used in several projects carried out by different companies, either public or private and big or small. A high number of Web systems with different providers, users or development teams are working with them.

Moreover, since 2004, an important project is being developed in liaison with the Andalusian Regional Cultural and Sport Ministry. This project made us realize that it was necessary to manage software quality and the set of processes for software quality assurance. We started to build NDTQ-Framework after detecting the necessity of defining these processes. However, an important problem appeared; if we only defined processes, the evolution would be complex. Normally, processes have to be changed or improved 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 Figure 1. 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 suitable tool to do it.

We had the chance of testing this solution in another project carried out in liaison with Emasesa. In this project, the testing phase had to be improved with the incorporation of ISO/IEC 29119 references. In this organization, NDT was used for the AQUA-WS project development (AQUA-WebServices) and the testing phase is especially critical in this project. In this context, we define the set of test processes presented in this paper.

Nevertheless, NDTQ-Framework is continually reviewed. Nowadays we are extending our approach with the aim of including the security processes defined by ISO/IEC 27000:2009.

7. Conclusions and Future Work

This paper presents an approach for software processes management. This approach, based on the definition of a business process metamodel, is implemented in a framework named NDTQ-Framework.

The paper starts with a global view of some standards and references that have to be considered by software companies. With this previous analysis and the definition and implementation of the metamodel, a global vision of the background of NDT is presented. The paper continues describing NDTQ-Framework in detail and finishes with some business references.

As a future work, we are reviewing our approach with the aim of improving it. As it was commented in the previous section, we are extending the approach to include security processes and some new tools are being developed in order to make easier the execution and control of these processes. For instance, NDT-Driver was improved to include test derivation and we also introduce some new rules in NDT-Quality for new processes and indicators.

Nowadays, we are developing a new tool to be added to NDT-Suite named NDT-Counter. It is a tool that, through the Use Cases Points technique, measures the effort of a project to be developed with NDT and it is oriented towards executing processes participating in project estimation.

Acknowledgements
This research has been supported by the Tempros project (TIN2010-20057-C03-02) of the Ministry of Education and Science, Spain (Ministerio de Educación y Ciencia), as well as the NDTQ-Framework project (TIC-5789) of the Andalusian Regional Government, Spain (Junta de Andalucía).

References:

OMG, "BPMN, Business Process Modeling Notation, Version 2.0" 2011. Available: www.omg.org/spec/BPMN/2.0/. Last Accessed 04-2013.

Pillain, P.-Y., Champeau, J.,Nhi Tran, H. "Towards an Enactment Mechanism for MODAL Process Models," in Proceedings of the 1st Workshop on Process-based Approaches for Model-Driven Engineering (PMDE), 2011.

OASIS, "WS-BPEL, Web Services Business Process Execution Language, Version 2.0". Available: http://www.oasis-open.org/standards#wsbpelv2.0. Last Accessed 04-2013.

Acuña, S.T., Juristo, N. Software Process Modeling, no. May. Springer Verlag, 2005, p. 232.

ISO/IEC 9001:2008 Quality management systems -- Requirements. International Organization for Standardization, 2008.

Chrissis, M. B., Konrad, M., Shrum, S. CMMI: Guidelines for Process Integration and Product Improvement. Addison Wesley, 2003, p. 688.

ISO/IEC 15504:2004 Information technology -- Process assessment. International Organization for Standardization, 2004.

ISO/IEC 12207:2008 Systems and software engineering - - Software life cycle processes. www.iso.org/iso/catalogue_detail?csnumber=43447. Last Accessed 04-2013.

ISO/IEC 15288:2008 Systems and software engineering - - Software life cycle processes. www.iso.org/iso/catalogue_detail?csnumber=43564. Last Accessed 04-2013.

Prikladnicki, R., Nicolas Audy, J.L., and Evaristo. R. 2006. A Reference Model for Global Software Development: Findings from a Case Study. In Proceedings of the IEEE international Conference on Global Software Engineering (ICGSE '06). IEEE Computer Society, Washington, DC, USA, pp. 18-28.

ISO/IEC TR 24774:2010 Systems and software engineering -- Life cycle management -- Guidelines for process description. Last Accessed 04-

2013. www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=53815

ISO/IEC 29119 Software Testing. The new international software testing standard. Last Accessed 04-2013 www. softwaretestingstandard.org.

Information Technology Infrastructure Library. Last Accessed 04-2013. www.itil-officialsite.com.

Information Technology Service Management. Last Accessed 04-2013 . www.itsm.info/home.htm.

Project Management Body of Knowledge. Last Accessed 04-2013.www.pmi.org/PMBOK-Guide-and-Standards.aspx

The Guide to the Software Engineering Body of Knowledge. Last Accessed. 04-2013 www.computer.org/portal/web/swebok

Software Process Improvement and Capability determination. Last Accessed 04-2013. www.sqi.gu.edu.au/spice/

IT Governance Institute. "COBIT 4.1". ITGI, 2007.

Pardo, C., Pino, F., García, F., Piattini, M., Baldassarre, M.T. " Trends in Harmonization of Multiple Reference Models in Evaluation of Novel Approaches to Software Engineering" LNCS (2011)

ISO/IEC 90003:2004. Last Accessed 04-2013. www.iso.org/iso/catalogue_detail?csnumber=35867.

ISO/IEC 27000:2009. Last Accessed 04-2013. www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=41933

Pardo, C., Pino, F., García, F., Piattini, M., Baldassarre, M.T, "HProcessTOOL: A Support Tool in the Harmonization of Multiple Reference Models" ICCSA 2011, Part V, LNCS 6786, pp. 370–382, 2011.

MÉTRICA. VERSIÓN 3. Metodología de Planificación, Desarrollo y Mantenimiento de sistemas de información. Last Accessed 04-2013. http://www.csi.map.es/csi/metrica3/index.html.

OMG. Unified Modelling Language: Superstructure, Version 2.0. Specification, Object Management Group. http://www.omg.org/cgi-bin/doc?formal/05-07-04, Last Accessed 04-2013.

Koch, K., Knapp, A., Zhang, G., Baumeister, H. UML-Based Web Engineering. Web Engineering: Modelling and Implementing Web Applications. Springer, 2008. pp. 157-191.

Ceri, S., Fraternali, P.,Bongio. A. Web Modelling Language (WebML): A Modelling

Language for Designing Web Sites. Conference WWW9/Computer Networks, 33(1-6), 2009. pp. 137-157.

Rossi, G., Schwabe, D. Modelling and Implementing Web Applications with OOHDM. Web Engineering: Modelling and Implementing Web Applications. Springer, 2008. pp. 109-155.

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.

OMG, "QVT, Meta Object Facility (MOF) 2.0 Query/View/Transformation". Last Accessed 04-2013. Available: http://www.omg.org/spec/QVT/1.1/.

NDT-Suite. Available in www.iwt2.org. Last Accessed 04-2013

García-García, J.A., Cutilla, C.R., Escalona, M.J., Alba, M., Torres, J. "NDT-Driver, a Java Tool to Support QVT Transformations for NDT," in the 20th International Conference on Information Systems Development (ISD), 2012, pp. 170-176.

García-García, J.A., Cutilla, C.R., Escalona, M.J., Alba, M. "NDT-Glossary. A MDE Approach for Glossary Generation," in Proceedings of the 13th International Conference on Enterprise Information Systems. ICCEIS, 2011.

Kruchten, P. The rational unified process: an introduction. Addison Wesley Professional, 2004.

M.J. Escalona, J.J. Gutiérrez, D. Villadiego, A. León, A.H. Torres. Practical Experiences in Web Engineering. Advances in Information Systems Development. New Methods and Practise for the Networked Society. Vol 2. Pp.421-434. 2007.

4UNE 166002:2006 Last Accessed 04-2013. www.aenor.es/aenor/normas/normas/fichanorma.asp?tipo=N&codigo=N0036136.