Practical Model Driven Development process

Xabier Larrucea, Ana Belen García Díez, Jason Xabier Mansell

European Software Institute
Xabier.Larrucea@esi.es, anabelen.garcia@esi.es, jason.mansell@esi.es

Abstract. Nowadays many organizations are adopting MDA to describe their systems. This fact forces organizations to transform their software development process into a Model-Driven Development process. This paper proposes a software development methodology focused on MDA, and describes both the MDD process as well as the main process workflow. The UML Profile SPEM is used to describe the process. In this paper we present a MDD process and a set of System Family Engineer concepts to adapt the MDD process according to user and functional requirements. This methodology has been developed in a European IST project (MASTER project IST-2001-34600)

Introduction

Many organizations have already realized that the UML usage is becoming more and more important to define their systems. In fact this modelling language is a core concept within the MDA (Model Driven Architecture [11]) standard defined by the OMG (Object Management Group). When these organizations put into practice the MDA philosophy, they need to adopt a Model-Driven development process and the appropriated tools to support it. This paper is focused in the MDD process.

Nowadays many software development processes (SDP) like RUP (Rational Unified Process) are being applied in the industry. However these processes are not taking into account MDA concepts and they must be fit into this context. Others SDP like XP (eXtreme Programming), are also being applied. This SDP is an agile method and therefore the design phase is code-oriented whereas MDA is model-oriented. In [4] Stephen J. Mellor et al. combine the notion of “agile” and “model” and other work related with processes has already been published, such as [8] and [7].

This paper presents a methodology developed in the MASTER project, a European IST project (IST-2001-34600). In this paper we present a MDD process and a set of System Family Engineer concepts to adapt the MDD process according to user and functional requirements. The process is described in SPEM [12] (Software Process Engineer Metamodel) notation. This paper completes the work presented in [6].

This paper is structured in three main sections; Section 2 provides an overview of the MDD process; Section 3 outlines the adaptative process. Finally section 4 concludes the paper with future research action lines.

The MDD process

Many software development processes are considered as heavyweight processes. Moreover processes like RUP (Rational Unified Process) could be adaptable to Model Driven Architecture. For example, in [5] Chris Raistrick et al. have demonstrated how MDA could be applied ("Using MDA in a typical project"). However their process is a heavyweight process, it's focused in eXecutable UML(xUML) formalism and they do not take account the architectural layers. Our main process could be also considered as a heavyweight process but with some differences. Our process is based on the different architectural layers defined to describe and model a domain [3]. These layers are well-defined through the different metamodels definition.

In this section the MDD process is defined outlining the different phases with a brief description. Each phase contains a set of activities that are deeply explained in MASTER project deliverables [2]. The phases and the activities are tightly related with PIM layers definition. The basis of the architectural layers are already described in others works [10].

Figure 1 and Figure 2 provide an overall picture of the methodology proposed. Figure 1 provides an overview of the phases of the methodology whereas Figure 2 provides a more detailed overview of the MDD process workflow, describing the work products required and derived in each phase of the methodology.

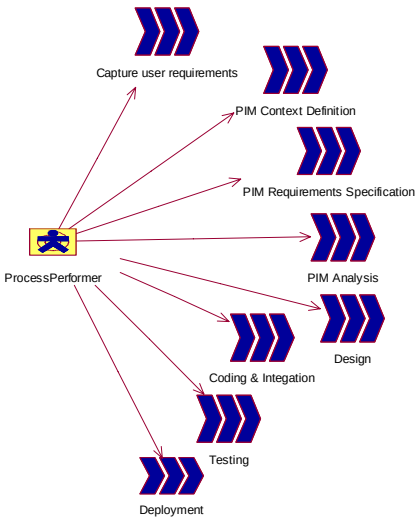

Figure 1 : Phases overview

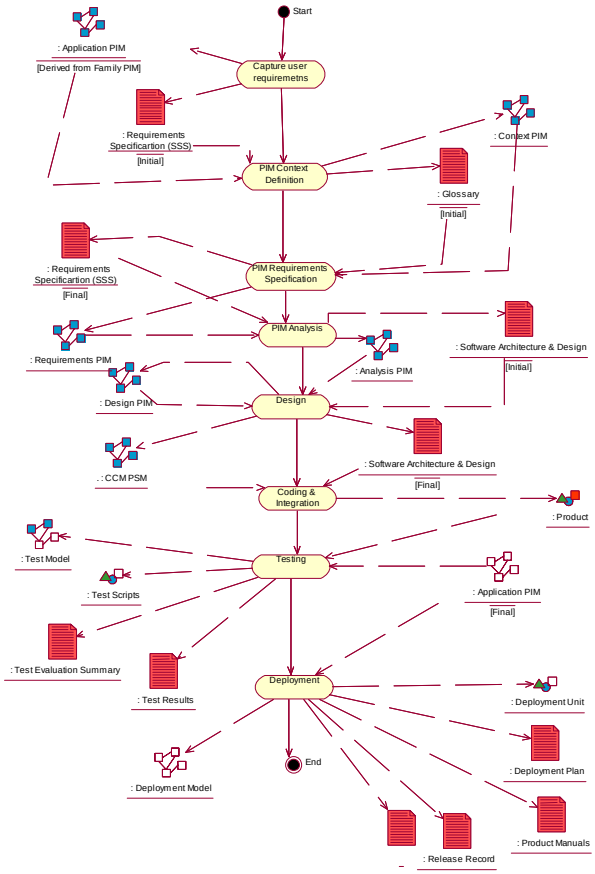

Figure 2: MDD process workflow

Figure 1 provides and overview of the phases that make up the methodology proposed. The phases are**:**

- **Capture User Requirements:** The objective of this phase is to elicit, agree and document the customer requirements that the software system needs to fulfill. This includes establishing a common understanding with the customer on functional and non-functional requirements. This phase includes the following activities: formalize the customer requirements in an Application Model and derive an initial Application PIM and an initial functional requirements specification from the common infrastructure of reusable assets.

- **PIM Context Definition:** The objective of this phase is to clearly define the scope of the software system to be developed. The result is an unambiguous definition of the system, its objectives, and scope following a black-box approach. Main activities are:
    - Establish the system goals and business principles.
    - Describe the external actors that interact with the system.
    - Identify the high-level services offered by the system and their key behaviour.
    - Define the business events, and exchanged business objects.
- **PIM Requirements Specification:** The objective of this phase is to build a model of customer requirements clear and complete and to have a unique requirements description that all subsequent models will use. In order to model the system functional and non-functional requirements, the main activities of this phase are:
    - Refine the PIM Context
    - Identify services, events and business objects produced and consumed by the system and the actors interacting with the system
    - Specify capabilities (use cases), forces (non-functional requirements), and atomic requirements
    - Identify and model the relationships between functional and non-functional requirements.
- **PIM Analysis:** The objective of this phase is to model the internal view of the system without any technological consideration and maintaining the separation of concerns between functional and non-functional aspects. The main activities of this phase are:
    - Describe the system functionalities: the objects (with classes, attributes, packages, etc.), the functions (with operations), the system boundary (with interfaces), the behaviour (with sequence diagrams), etc.
    - Describe the system QoS aspects (refine the classes) and their application to the functional elements of the model.
    - Maintain traceability with the Requirements PIM.
- **Design:** The objective of this phase is to model the detailed structure and behaviour of the solution (software application) that fulfils the system functional and non-functional requirements. This implies making decisions on how the system will be implemented and which architectural style, patterns, standards and platforms will be used. Following an MDA approach, the design is performed in two steps:

- Specify and design a platform-independent solution (how) for all the requirements (what). The PIM will be defined with different elements depending on the architectural style selected for the solution, e.g., for a Components Design PIM the solution is expressed in terms of software components (component, interface, port, connector).
- Specify and design the platform-specific solution by refining the platform-independent solution. The PSM is intended to be automatically derived from the PIM through transformation engines. The PSM contains models specific of the platform (e.g., CCM, EJB, .NET) and is detail and complete enough to allow the codification and deployment of the solution

- **Coding & Integration:** The objective of this phase is to develop and verify the software code that implements the software design fulfilling the software requirements. This phase includes activities such as: develop the components and classes (according to the models used as inputs), define the organization of the code, execute unit tests, and integrate components and subsystems. Following a MDA approach, the code is intended to be automatically produced from the PSM through transformation engines.

- **Testing:** The objective of this phase is to demonstrate that the final software system satisfies its requirements. This phase includes activities such as: plan tests, prepare test model, test cases and test scripts, execute tests, correct defects and document testing results. Test models are traceable to PIM models (specially to PIM Requirements) and, following an MDA approach, test models will be refined from the PIM and test cases and test scripts will be automatically produced from the test model through transformation engines.

- **Deployment:** The objective of this phase is to ensure a successful transition of the developed system to the final users (including resources, environment, schedule planning and execution). This phase includes activities such as: create a deployment plan (dates of installation, resources, etc.), create a deployment model (derived from the PSM Deployment model and adapted to the specific execution environment of the customer), create the product manuals, maintain records of the product that is being delivered to the client, and provide the installation of the product in the client premises

In this Model Driven Development process a set of roles are also described. Moreover each phase is described through a workflow diagram in SPEM notation. The purpose of this paper is not to give a deep and exhaustive description of the elements of the entire process. However these elements are described in the MASTER deliverables [1] and [2].

Adaptative Process

In the previous section a MDD process is described. This process is shown as standard software process (SSP). Many organizations adapt their SSP to their specific needs and requirements to provide software development plans. These plans have a *set of items that have common aspects and predicted variabilities* [9] (a process family). Therefore System Family Engineering concepts can be applied in this domain. The MDD process could be adapted to user requirements establishing relationships between application models (the MDD process and functional model).

Figure 3 provides an overview of the system familiy engineering process, in which based on a detailed analysis of a domain, a set of decisions can be defined which identify univocally any product of a domain. These set of decisions are captured in a decision model which captures the variability of a domain. Once this variability is solved by using the user requirements, an application model is produced. This application model captures user requirements and is used in order to inititate the derivation process by transformations in which the specific requirements are introduced within the derivation process and the application variabilities are resolved. As a result of the derivation process, in which all variabilities within a domain are solved, the application assets for a specific customer are produced.

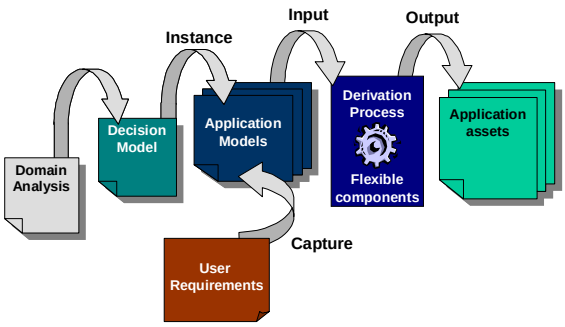

Figure 3: System Family Engineering overview

The main purpose of this paper is not describe how the MDD is produced step by step but how the MDD process described in the previous section is tailored with user needs and how some activities are removed or added depending on the requirements (derivation process). This customization process is defined through variability management, described in Figure 3. Within ESI a tool suite called V-Manage is used to define and to implement the variability of the MDD process.

Figure 4 provides an overview of how Model Driven Engineering and System Family Engineering have been used to produce a MDD adapted process.

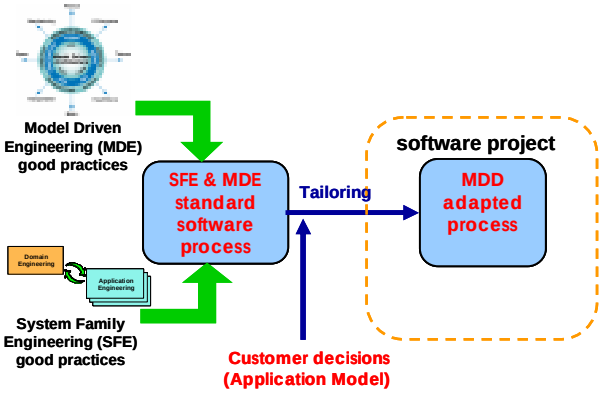

Figure 4: MDD adapted process

Conclusions and future work

This work has been developed in a European IST project called MASTER project (IST-2001-34600) and it has also been applied in the context of Air Traffic Management jointly with Thales ATM.

In this paper a Model Driven Development process has been described. Some parts of this methodology like roles and work products description have been omitted to limit the size of the paper. Moreover SFE has been applied to take into account user requirements to customize the general process. However to complete the overall process an appropriate tool suite must to be provided. Actually, it would be suitable to have an IDE (Integrated Development Environment) supporting the domain analysis phase throughout the deployment phase. Many tool vendors have MDD compliant tools but do not provide support for the overall process or do not provide features such as non-functional aspects (Rational XDE Modeller) related with behavioral features.

In the context of methodology an emergent initiative related with MDA, agile modelling (AM) [13] is growing. However tool vendors must improve their tools to be able to execute models. Methodologies related with this “agile” area will be the focus of our future work.

Reference:

[1] Deliverable D3.1 "*Enriched PIM with project management information*". MASTER project: IST 34600. (http://modeldrivenarchitecture.esi.es/mda_publicDocuments.htm#D3.1)

[2] Deliverable D3.2 "*Process model to engineer and manage the MDA approach*". MASTER project: IST 34600. (http://modeldrivenarchitecture.esi.es/mda_publicDocuments.htm#D3.2)

[3] Deliverable D2.1 "*PIMs Definition and Description to model a domain*". MASTER project: IST 34600. (http://modeldrivenarchitecture.esi.es/mda_publicDocuments.htm#D2.1)

[4] *MDA Distilled. Principles of Model-Driven Architecture*. Stephen J. Mellor, Kendall Scott, Axel Uhl, Dirk Weise. Addison-Wesley. Series Editors. Object Technology Series

[5] *Model Driven Architecture with Executable UML*. Chris Raistrick, Paul Francis, john Wright, Colin Carter, Ian Wilkie. Cambridge

[6] *Process Engineering and Project Management for the Model Driven Approach*. Ana Belen Garcia Diez, Xabier Larrucea. First European Workshop Model-Driven Architecture with Emphasis on Industrial Applications , Enschede, the Netherlands , March 17-18 2004

[7] *Application of MDA for the development of the DATOS Billing and Customer Care System (Case study on the use of MDA for the development of a larger J2EE System)*. Jorg Guther, Chris Steenbergen. First European Workshop Model-Driven Architecture with Emphasis on Industrial Applications , Enschede, the Netherlands , March 17-18 2004

[8] *Towards an MDA-based development methodology for distributed applications*. Anastasius Gavras, Mariano Belaunde, Luis Ferreira Pires, Joao Paulo A. Almeida. First European Workshop Model-Driven Architecture with Emphasis on Industrial Applications , Enschede, the Netherlands , March 17-18 2004

[9] *Software Product-line Engineering. A family based software development process*. David M.Weiss,Chi Tau Rober Lai. Addison-Wesley

[10] *PIM Definition and Description*. Daniel Exertier, Benoit Langlois, Xavier Leroux. First European Workshop Model-Driven Architecture with Emphasis on Industrial Applications , Enschede, the Netherlands , March 17-18 2004

[11] *MDA Guide v1.0.1*. Object Management Group, omg/03-06-01, June 2003

[12] *Software Process Engineering Metamodel v1.0 (SPEM)*, Object Management Group, formal/02-11-14 November 2002

[13] *Agile modeling* http://www.agilemodeling.com