RESEARCHERID
THOMSON REUTERS

[Samba, 4(7): July 2017] ISSN 2348 – 8034
DOI- 10.5281/zenodo.822733 Impact Factor- 4.022

GLOBAL JOURNAL OF ENGINEERING SCIENCE AND RESEARCHES

TAILORING AND INSTANTIATION OF MDE PROCESS MODELS: A Y MODEL-BASED APPROACH

Samba DIAW[*1], Mamadou Lakhassane CISSE[2] & Alassane BAH[3]
[*1,2&3]Polytechnic Institute (ESP) @ Cheikh Anta Diop University (UCAD), UMMISCO Laboratory
Dakar, Senegal

ABSTRACT

The adoption of MDE in software development has drastically changed the way of developing software products. It gives a way of automating software development while reducing production time. While software companies need continually to improve and customize their software processes, a quick approach to do so is still lacking. Most of those companies have an organizational process that is used whenever they have an upcoming development project. Reusing the same process for any development project is somehow inadequate. So, tailoring of such a process is necessary to fit projects specificities. However, even if that tailored process can be used for a specific project, it still lacks information about resources needed for execution. In this article, we propose a Y model-based approach that allows tailoring a MDE process model and instantiating it for a specific project in the context of an organization. This approach is made possible by using models transformations. We defined metamodels to express the context and resources of a project. It results in a more optimal use of stakeholder's roles and a better way of producing software. We illustrate our approach with the UWE Process in a context of a maintenance project with J2EE.

Keywords*: Model-Driven Engineering (MDE), Model Transformations, MDE Software Process, Tailoring, Instantiation*

I. INTRODUCTION

Nowadays, software applications play a crucial role in our life. Everywhere software applications are replacing men more and more. However, their implementation becomes more complex depending on their nature. To develop a software with a high quality we should have a good software process. Therefore, it is important, for companies to have their own generic process that can be adapted (i.e. tailor) to any project/organization context. To master software development, it is important to capture all changes that happen during its elaboration meaning the description of the actual (i.e. really) process.

According to Curtis et al, a software process model (process model, for short), is defined as "an abstract description of an actual or proposed software process which represents selected process elements that are considered important to the purpose of the model and can be enacted by a human or machine" [16]. A process model is an abstract representation and does not capture concrete information on how the model-products are actually managed during process execution.

To describe the software processes, several process-modeling languages (PML) were proposed (SPEM [17], XPDL [18], BPMN [18], UML4SPM [19], SPEM4MDE [2], eSPEM [20], xSPEM [21], CM-SPEM [23], etc.). However, none of them has gained much attention from the software community. These languages although allowing to specify the software processes (SPEM, BPMN, etc.) and fulfilling the executability criterion (XPDL, eSPEM, xSPEM) became quickly obsolete when they had to take into account the unexpected changes within a software development project.

So, to favor a better use of the software processes in the software development, the vision of the "model of process as execution plan" must be relegated in the background, so that gets the upper hand, the vision of the "model of process as abstraction of the reality of a software development project". These facts are changes which could have

been planned (the planed process model) or been unexpected (in the execution of the process model). Dice then their consideration becomes inevitable in the elaboration process of a software.

In our previous works [2][22], we proposed SPEM4MDE, a MDE PML based on the standard SPEM to describe the MDE processes. The objective of SPEM4MDE was (1) to propose an extension of SPEM in which the central concepts of the MDE processes are reified; (2) to propose a language dedicated to the behavioral modeling of the processes by means of state-machines; (3) to propose a conceptual architecture of an environment for MDE software processes modeling and enactment.

The interest of MDE concepts reification is to allow on one hand to the process designers to clarify specific aspects of the software development in a MDE context, and on the other hand to better insure the coherence of the models produced in such a process.

The integration of the behavioral modeling in SPEM4MDE allowed describing the life cycle of the elements of a project through state-machines. However, in spite of the fact that SPEM4MDE answers the issue of the integration of MDE in the software processes and their executability, its experiment showed some weakness.

For example, the lack of support to the process designers to automatically adapt a MDE process according to a project in an organization context is still missing. Another lack is the difficulties of taking into account the changes which inevitability happen in the execution of a software process (e.g. if a person gets ill it is necessary to replace him/her, the addition of a new utilitarian class, etc.).

To overcome those obstacles, we propose in this article, a Y model-based approach to tailor and instantiate MDE processes. To validate our approach, we are going to use the UWE process and apply it to a maintenance project in J2EE.

The remainder of the article is structured as follows: the section 2 presents our Y model-based approach while the section 3 presents its validation. The section 4 deals with related works. In the last section, we conclude this article and introduce some perspectives.

II. OUR Y-MODEL BASED APPROACH

Our contribution will consist in designing and implementing a Y model-based approach (seeFigure 1) to produce project specific process model (PSPM) from a generic process model so-called PIPM (Project Independent Process Model). This will be the first step of our approach and is a *tailoring transformation*. The second step will be the instantiation of the PSPM according to the project resources. As a result, an enactable process model will be produced and will include the actors/tools and tasks to be executed.
The Y model-based approach will involve four metamodels:

- a metamodel taking into account the concepts of software development process definition (SPEM4MDE)
- a metamodel that defines the context model of a project/organization (SPCM),
- a metamodel to represent the resources of any given project (to be defined)
- a metamodel that defines managed concepts at execution time (to be defined)

For the first part, the main idea is to consider generation as models transformation that takes two input models:

- a generic software development process model independent from any project so called PIPM (Project Independent Process Model) conforms to SPEM4MDE
- a model representing particular development project specificities so called PSM (Project Specific Model) conforms to SPCM (Software Process Context Metamodel).

The second step of our approach will include the project resources model and the result of the first transformation (i.e. tailored process model). This step produces an enactable process model.

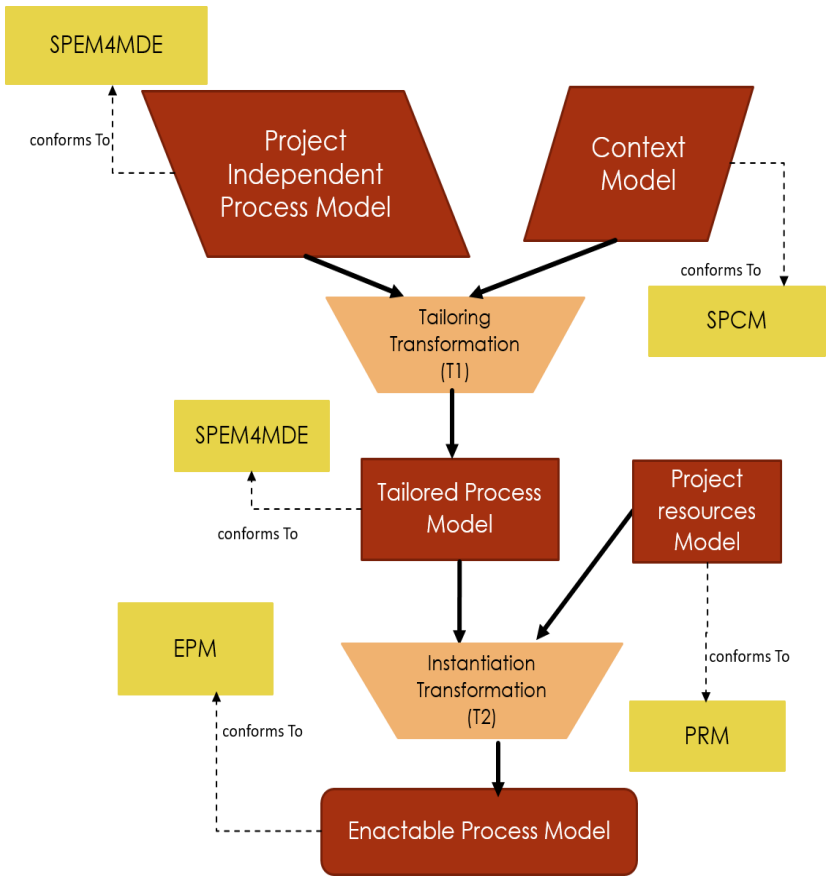

Figure 1. The Y model-based approach

a. Project Independent Process Model

To define the project independent process metamodel, we have decided to reuse SPEM4DME. SPEM4MDE [2] extends SPEM capabilities to take into account any kind of processes including MDE ones. It also proposes a MDE process behavioral modeling dedicated language to assist stakeholders, giving them at any moment their activities and their related states. SPEM4MDE introduces some packages: *MDE ProcessStructure*, *Model Relationship* and *MDE Process Behavior*. Reusing SPEM4MDE in our approach will largely extend processes support capabilities.

[Samba, 4(7): July 2017] ISSN 2348 – 8034
DOI- 10.5281/zenodo.822733 Impact Factor- 4.022

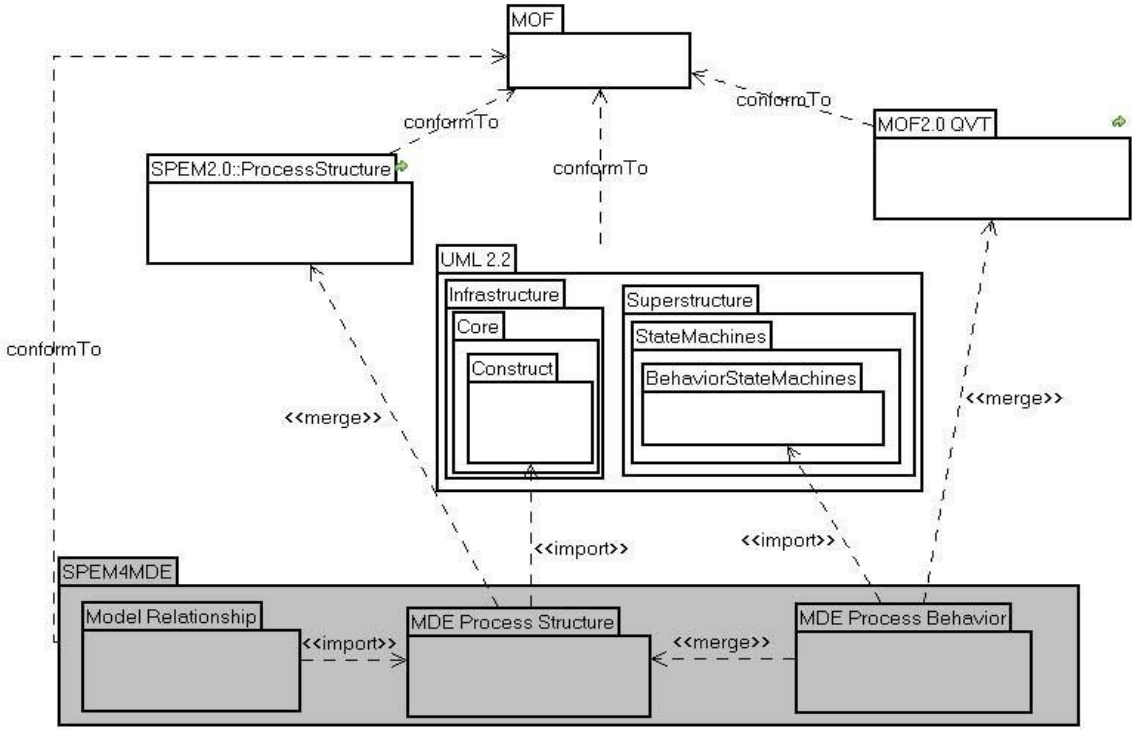

Figure 2. SPEM4MDE packages organization [2]

b. Context Model

The context model allows us to take decisions about variations in the PIPM. It describes the characteristics of a project that may vary according to a context allowing us to tailor automatically processes on the basis of those values. To represent the specificities of a project in our approach, we have reused an already defined metamodel for that purpose (i.e. SPCM [4]).SPCM is based on three main concepts: "*ContextAttribute"*, "*Dimension"* and "*ContextAttributeConfiguration*". The different characteristics of the process are represented by a *ContextAttribute* and their values by *ContextAttributeValue*.

Nevertheless, we are going, in our implementation, to add more precision in the way of defining thoseattributes values. For example, instead of defining variables project size as medium, large or small, we are going to give a numeric value on which the choices of tailoring are going to depend on. Thus, for the *ContextAttribute* Project size, we might have as values: 2, 4 or 10 people. Those values, more objective than those proposed in [4] enable us to defined scales in which the decision of tailoring is taken.

[Samba, 4(7): July 2017] ISSN 2348 – 8034
DOI- 10.5281/zenodo.822733 Impact Factor- 4.022

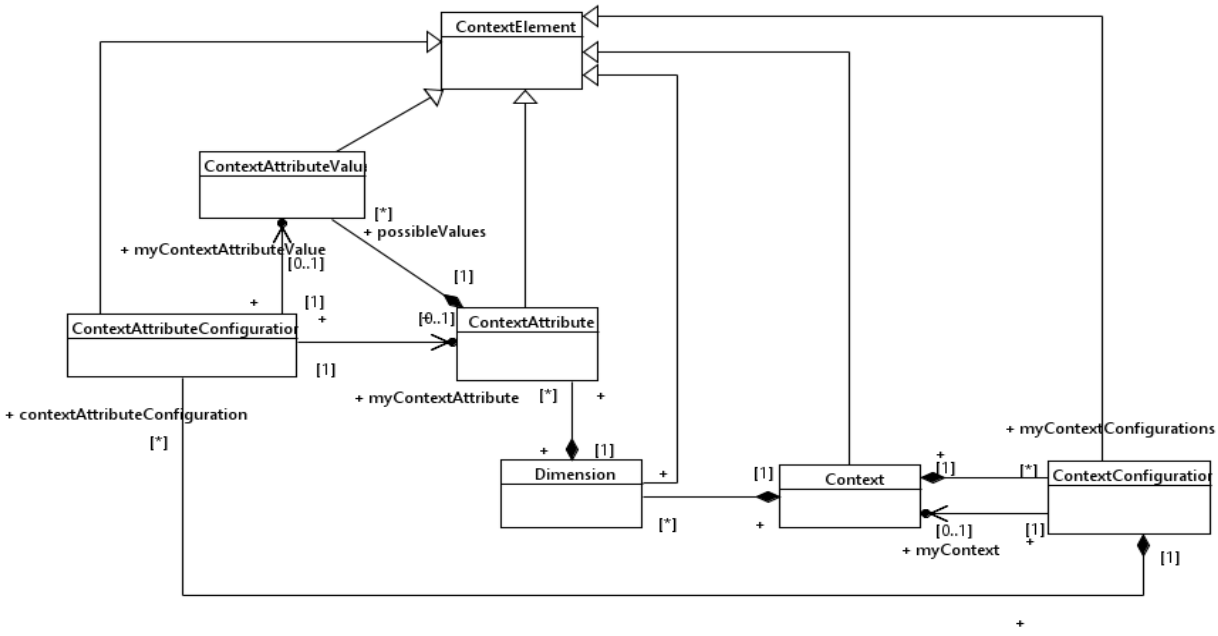

Figure 3. Software Process Context Metamodel – SPCM[4]

c. Project Resources Model

The resources used in a software development project are dependent of the project variabilities. The formalization of those elements enables the automatic instantiation of the tailored process. We have defined a new metamodel called PRM (Project Resources Metamodel) to define the resources model for any project. This metamodel describes every resource element within a software development project.The major concepts of PRM are:*ResourceElement* and *ResourceType*. A resource element defines every resource used within a software development project (actors, tools, etc.). It has a name, a description and a kind (which is an enumeration of *ResourceKind*) defining if the resource is primary or secondary. The *ResourceType* conceptdefines the type of resource, which can be human, software, or hardware. Resource typesmight not have the same properties. For solving that issue, we defined the *ResourceProperty* concept in order todefine new properties for a *ResourceType*. Resource elements can be organizedin a group or team and are linked with their role (human resources only).Examples of *ResourceElement* can be Bob (name), which is a human *ResourceType.* For the human*ResourceType* we can also define different properties: address, phone number, family name. Note that the list is not exhaustive and give the responsibility to the project manager to define new suitable properties.

[Samba, 4(7): July 2017] ISSN 2348 – 8034
DOI- 10.5281/zenodo.822733 Impact Factor- 4.022

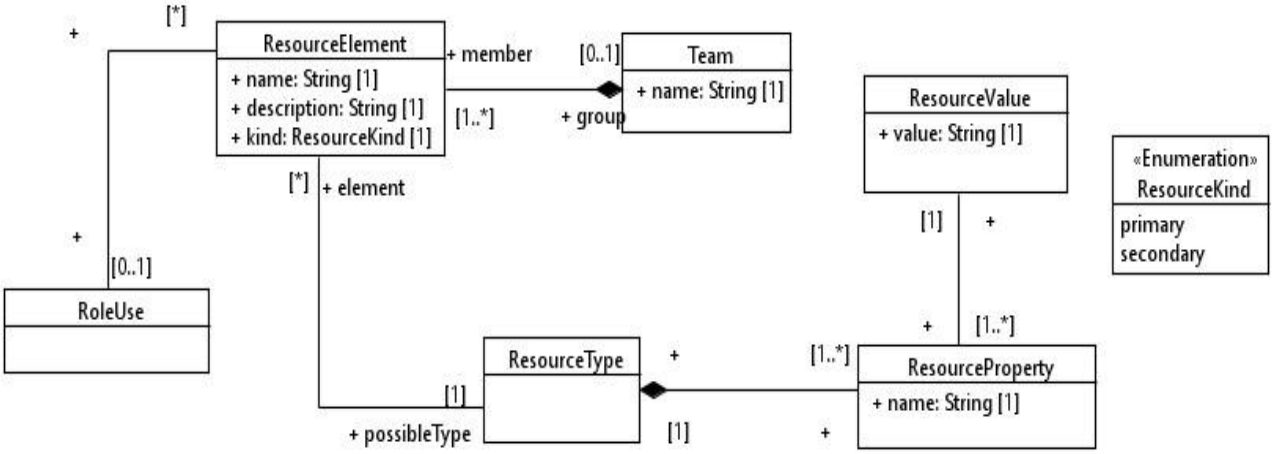

Figure 4.Project Resources Metamodel - PRM

d. Tailored Process Model

As a result of the tailoring process, we obtain a tailored process model. It is a project specific process model meaning it has the only activities that are necessary for that specific project. The tailored process model also conforms toSPEM4MDE metamodel but with a more precise perimeter of the project activities. All activities that are not part of the project context are ignored or removed.

e. Enactable Process Model

The instantiation process will produce a model ready to be enacted. The importance of this model is to give to the project manager all the information abouttasks and their performers. The model will contain the different resources chosen by the project team for taking part in the execution. The resources are not only human-like but any kind of resourcethat will participate in the production of the real artifacts. Having a metamodel representing these concepts is a major key of our approach and enables us to capture the real process (i.e. process in life).For that purpose, we have defined a new metamodel called EPM (Enactable Process Metamodel). The main concepts of EPM are *TaskExecution*, *MDEActivityInstance* and *TransformationExecution*. They help defining the activities and tasks that are to be executed. *TransformationExecution* is also a *TaskExecution*. The *TaskExecution*concepttakes as parameters one or many *ModelInstance (i.e. concrete model-products)*. Every activity or task is considered as an *enactment element* meaning that they are the elements that are going to be executed. We also reused the*ResourceElementconcept*from the Project Resource Metamodel to represent the specific resource for executing a task or transformation.

[Samba, 4(7): July 2017] ISSN 2348 – 8034
DOI- 10.5281/zenodo.822733 Impact Factor- 4.022

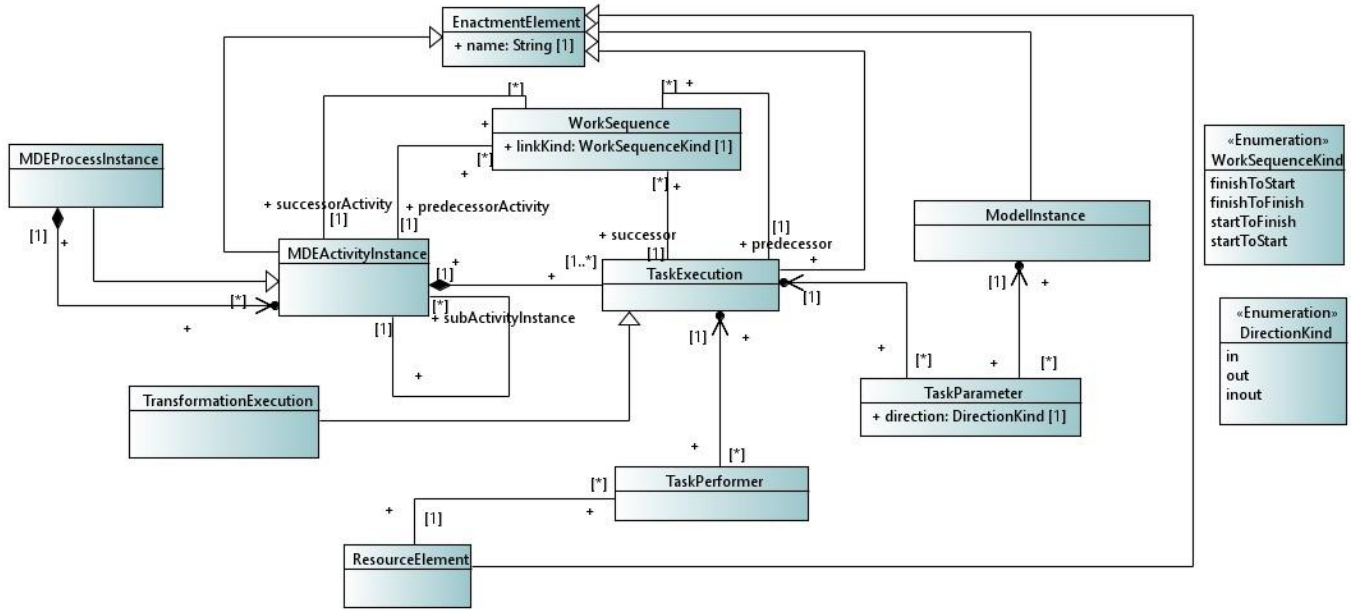

Figure 5. Enactable Process Metamodel

III. VALIDATION

To illustrate our Y model-based approach, we have chosen as an example the UWE (UML-based Web Engineering) process[24][25]. UWE is a process that covers web systems development cycle from requirements to code generation. The figure 6 represents the UWE process described with SPEM4MDE.

- **Our Project Independent Process Model (UWE Process)**

The first activity of our process is the description of the requirements model (*Describe Requirements Model*). It is performed by a *Web Designer* and produces a *Requirements Model*. This output is used for the following activity (*Requirements 2 Functional*) to produce *Functional Models.Functionalmodels* are afterwards integrated mainly for the purpose of verification into a *bigpicturemodel*. A merge of this *bigpicturemodel* with the *architecturalmodels* results in an *integratedmodel* covering functional and architectural aspects. Finally, platform specific models (PSM) are derived from the *integratedmodel* from which programming code can be generated. The transformation of *requirementsmodel* into *functional models* is a composite *transformation*, each sub-transformation dealing with a separate concern of Web engineering.

[Samba, 4(7): July 2017] ISSN 2348 – 8034
DOI- 10.5281/zenodo.822733 Impact Factor- 4.022

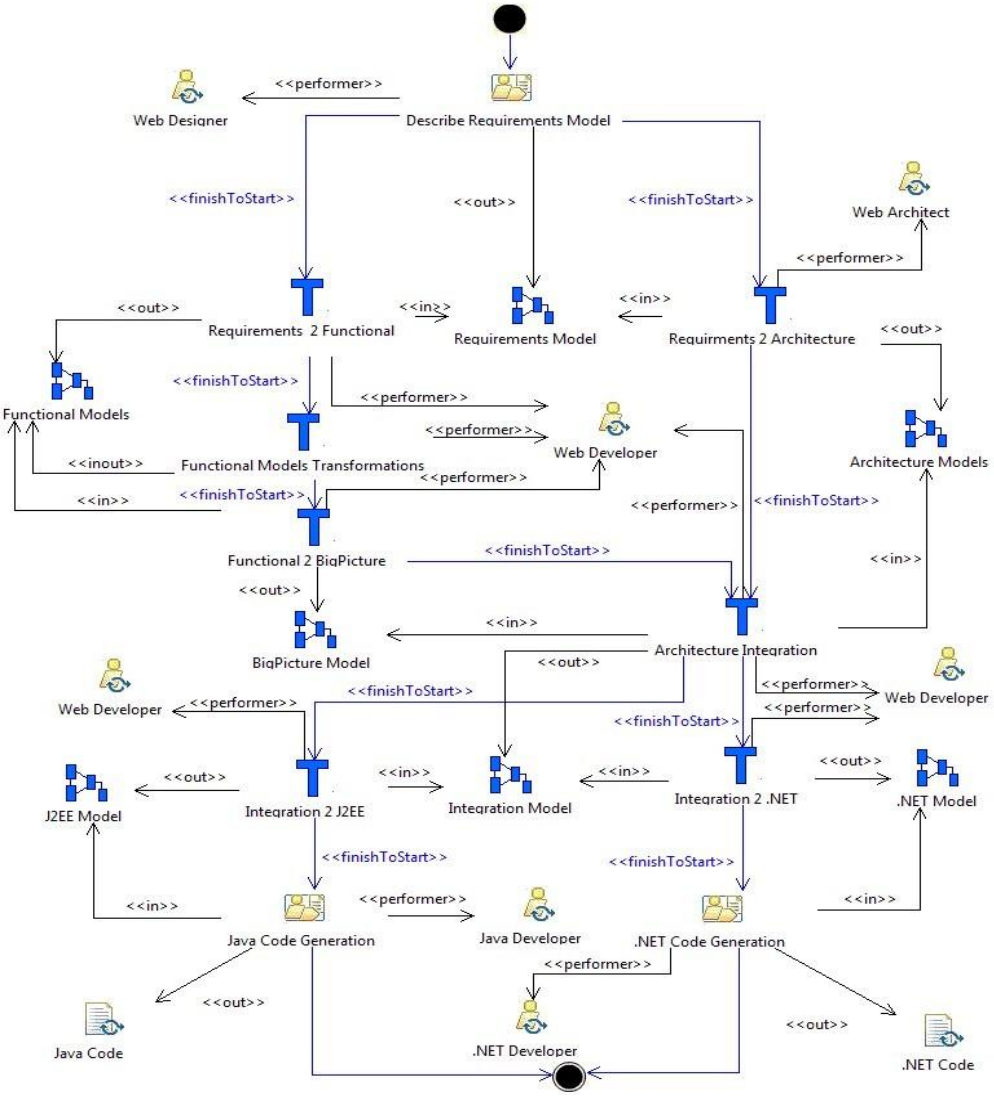

Figure 6. UWE Process structural description

- **Our Context Model**

To tailor the UWE process, we must have a context model representing process variations. This context model will define the specific characteristics we have chosen to deal with the process. In this way, we can configure new process models through models' transformation. The characteristics of the specific project are provided by the project manager. This will result in the generation of a new adapted process model.

The context variables considered in this tailoring process are the *project type* and the *deployment environment*. SPCM allows us to create more variables but we rather stick to these two variablessince they describe enough our context model.

Table 1: Context elements and values

Context Attribute	*Value*
Project type	*Maintenance execution*
Deployment environment	*J2EE*

Table I gives the values for our two context attributes. The tailoring process that is done based on them will give a new process adapted to the context of the project. We use ATL[26] to define the tailoring transformation rules.

- **The Tailored UWE Process Model**

The execution of the tailoring transformation (T1) allows us to configure a new process. That process will be adapted to the project context and is obtained through automatic generation. Figure 7 represents the resulting process after the tailoring activity. Only the required activities roles and artifacts are present. The obtained process does not include any additional activity. The "architecture integration" activity is removed, as it is not mandatory in the execution of a maintenance project. The "integration to .NET"transformation and ". NET code generation"activity arealso removed.

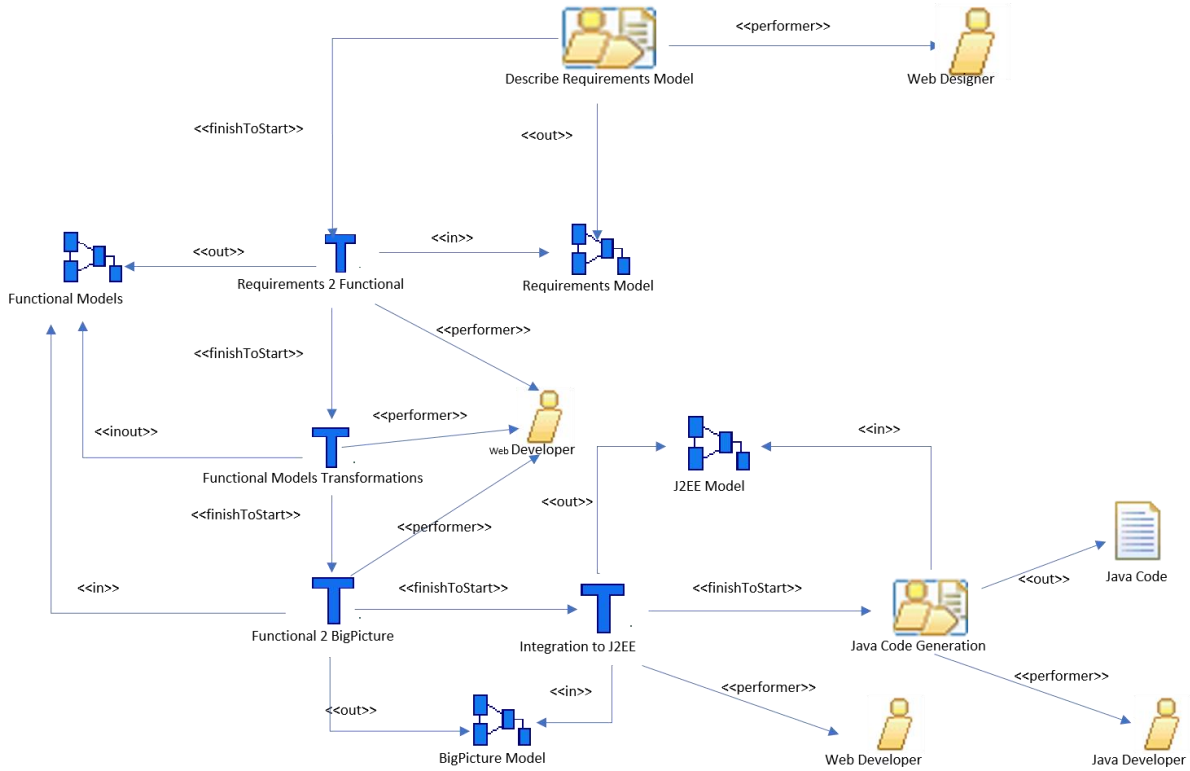

Figure 7: Tailored UWE process structural description

[Samba, 4(7): July 2017] ISSN 2348 – 8034
DOI- 10.5281/zenodo.822733 Impact Factor- 4.022

- **Our Project Resources Model**

In our Project Resources Model, we will give the effective resources in charge of tasks execution. In the UWE process, human actors do some activities whereas transformations are executedby MDE tools. The involved roles in the UWE process are:

- Web Designer
- Web Developer
- Java Developer

To instantiate our tailored process model, we are going to choose real actors for those roles. For the role Web Designer, we can have ahuman resourcewith its different properties: firstname, lastname, address. Our Web Designer will then, be named: Bob. The Web Developer and Java Developer are also human roles and will be respectively played by: Alice and Trevor.

- **The resulting Enactable Process Model (Enactable UWE Process)**

Once we have the tailored process, we can go through an instantiation with the real actors. The instantiation strategy enables us to have a final process model so-called enactable process model. This process model contains tasks to be executed and also the real actors having to execute them (Bob, Alice, Trevor). It still conforms to the EPM metamodel. Figure 8 shows the last step of our approach that produces the enactable UWE process.

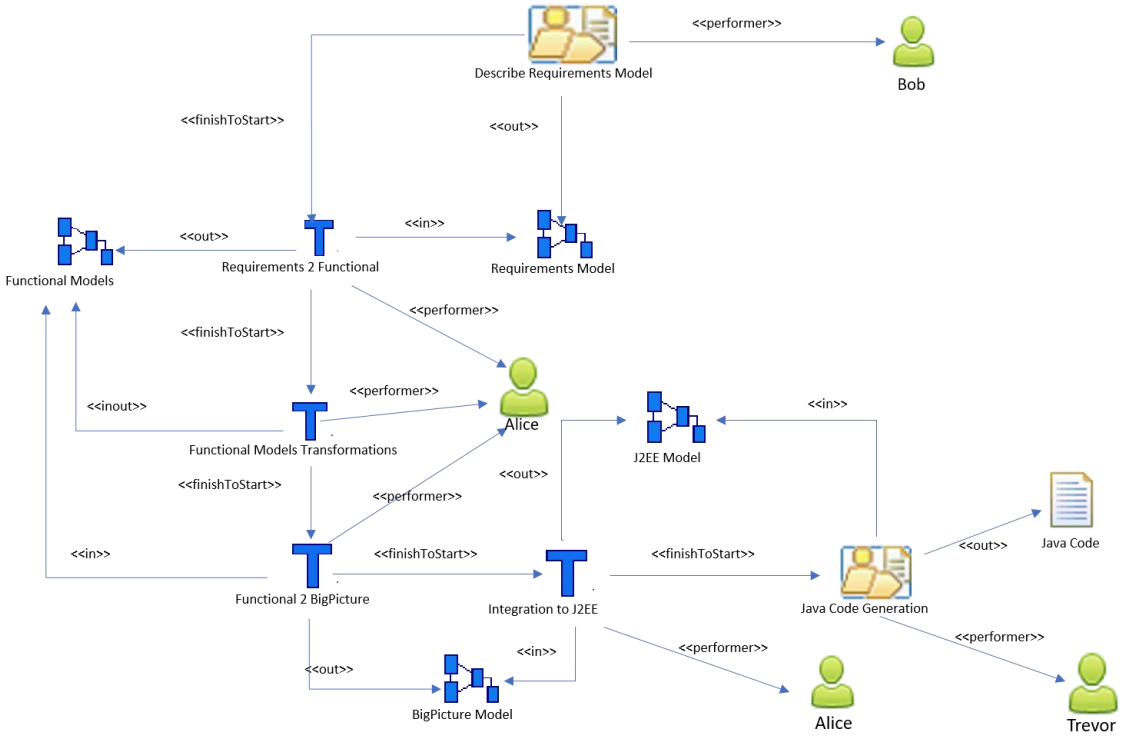

Figure 8: Enactable UWE Process

IV. RELATED WORKS

Process tailoring is the mechanism of adapting a software process to project needs[5].

Recent researches[6] have shown that different tailoring methods have been used over time. In some cases, it is done on the organizational level and in others on the project level. Tailoring on the organizational level allows adapting a standard process to the needs of a specific organization. The resulting process is adapted to the needs of each individual company.

Considering that projects in a single company can also differ, we need to tailor process at the project level, which means that the resulted process of the organizational level is adapted to the needs of a specific project.

Some work done around tailoring is[7], which presented a simple pragmatic method for adapting RUP to a specific project type in a company. They report that in their experience, process tailoring in small companies is best done as a simple and pragmatic process, and not as one, which is over-extravagant and strict. In[8], a set of agile practices tailored for large-scale complex projects has been proposed.

In[9], an example of template-based tailoring is presented. For each possible project situation, a well-defined process is established answering a scenario. For every scenario that might occur, one of the defined processes is chosen and executed for software development. This method is also used in[10] by taking into account project criticity and team size to choose the right process. This type of approach is highly depending on a complete knowledge of projects type and size that the company will have to deal with.

Using criteria to be applied in the tailoring process is an important task. However, those criteria must be carefully chosen to see which ones influence more the tailoring process[11] and even the links between those criteria. Furthermore, each criterion has its impact on a specific kind of project and none on another one.

In [12], a set of measures is provided to take into account different project situations. For each criterion, they ask two questions: What does it means (rationale)? and what might happen when not considering this particular criterion (implication)?

Another attempt in[13] focuses on the requirements for tailoring software processes. Unlike the former paper, they did not show up concrete criteria but focus more on the elements being used for tailoring and the causes of variations during process tailoring.

One of the common criteria find in the literatures is the team or company size. The project type also is one the most shared criterion for process tailoring.Among the factors that influence the software process we have the project, the organization, the product and the stakeholders of the project[12]. Figure 9 shows four major steps in software process tailoring.

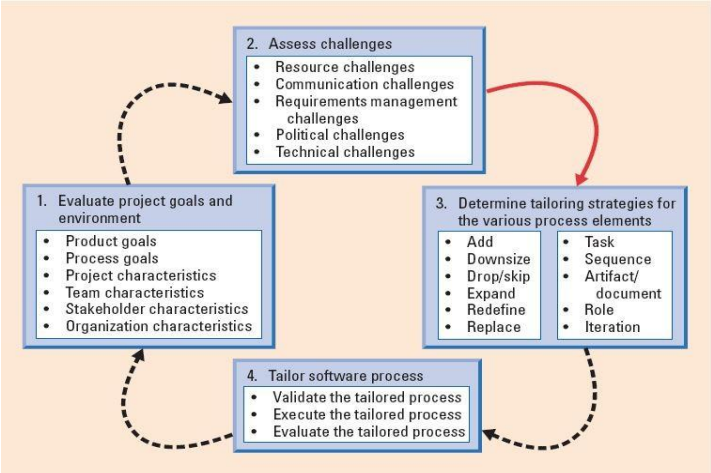

Figure 9: Software process tailoring steps[12]

The criteria can also be split in four groups [14]:

- the ones with regard to the *organization*,
- the ones with regard to the *project*,
- factors related to the *product*,

[Samba, 4(7): July 2017] ISSN 2348 – 8034
DOI- 10.5281/zenodo.822733 Impact Factor- 4.022

- factors related to human agents.

In[15], a model-based approach to software process tailoring has been proposed (fig. 10). Even if the proposal approach has been applied for a medium-size Chilean company, the concepts employed will not entirely change when applied to a larger company. This approach is made possible using organizational process model conforms to SPEM and a project context model. The transformation rules written in ATL will accordingly to the metamodels, produce an adapted project software process model still conforms to SPEM.

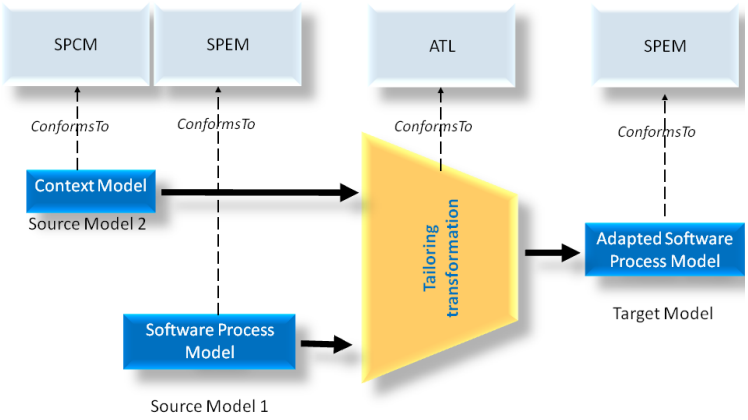

Figure 10: An MDE approach to tailoring [15]

V. CONCLUSION

In the literature, a few PML (Process Modeling Language) are natively supported MDE concepts. Reification allows promoting MDE concepts as first-class citizens.

In this paper, we have presented a Y model-based approach that allows tailoring and instantiating MDE processes. Our approach involves two main activities tailoring and instantiation. We use an automated support to assist process designer in those two complex activities.

We validate our approach with the UWE process, which we adapt withina context of a maintenance project in J2EE. The tailored process is instantiated with project resources in order to produce an enactable process model.

Two important perspectives of this work are under consideration. Firstly, we plan to develop a process engine to assist stakeholders in the execution of their tasks. Secondly, we envisage defining a full collaborative process execution metamodel for the enactment purpose.

REFERENCES

1. *C. González Pérez and B. Henderson-Sellers, Metamodeling for software engineering. Chichester: Wiley, 2008.*
2. *S. Diaw, R. Lbath, and B. Coulette, "Specification and Implementation of SPEM4mde, a metamodel for MDE software processes." in SEKE, Miami - USA, 2011, pp. 646–653.*
3. *O. M. G. Specification, "OMG unified modeling language (OMG UML), Superstructure, V2.4.1," Object Management Group, 2007.*
4. *J. A. Hurtado Alegra, M. C. Bastarrica, A. Quispe, and S. F.Ochoa, "An MDE approach to software process tailoring," in Proceedings of the 2011 International Conference on Software and Systems Process. ACM, 2011, pp. 43–52. [Online]. Available:http://dl.acm.org/citation.cfm?id=1987885*
5. *L. Silvestre, M. C. Bastarrica, and S. F. Ochoa, "A model-based tool for generating software process model tailoring transformations," in ModelDriven Engineering and Software Development (MODELSWARD), 2014 2nd International Conference on. IEEE, 2014, pp. 533–540.*

6. *O. Pedreira, M. Piattini, M. L. Luaces, and M. R. Brisaboa, "A Systematic Review of Software Process Tailoring," vol. Volume 32 Issue 3. New York, NY, USA: ACM SIGSOFT Software Engineering Notes, May 2007, pp. 1–6.*
7. *G. K. Hanssen, H. Westerheim, and F. O. Bjørnson, "Tailoring RUP to a defined project type: A case study," in International Conference on Product Focused Software Process Improvement. Springer, 2005, pp. 314–327.*
8. *L. Cao, K. Mohan, P. Xu, and B. Ramesh, "How extreme does extreme programming have to be? Adapting XP practices to large-scale projects," in System Sciences, 2004. Proceedings of the 37th Annual Hawaii International Conference on. IEEE, 2004, pp. 10–pp.*
9. *F. González, L. Silvestre, M. Solari, and M. C. Bastarrica, "Template Based vs. Automatic Process Tailoring," in XXXIII International Conference of the Chilean Society of Computer Science (SCCC 2014), 2014.*
10. *A. Cockburn, Crystal clear: a human-powered methodology for small teams. Pearson Education, 2004.*
11. *G. Kalus and M. Kuhrmann, "Criteria for software process tailoring: a systematic review," in Proceedings of the 2013 International Conference on Software and System Process. ACM, 2013, pp. 171–180.*
12. *P. Xu and B. Ramesh, "Using process tailoring to manage software development challenges," IT Professional, vol. 10, no. 4, pp. 39–45, 2008.*
13. *T. Martínez-Ruiz, J. Munch, F. García, and M. Piattini, "Requirements and constructors for tailoring software processes: a systematic literature review," Software Quality Journal, vol. 20, no. 1, pp. 229–260, Mar. 2012.*
14. *N. Du Preez, D. Lutters, and H. Nieberding, "Tailoring the development process according to the context of the project," CIRP Journal of Manufacturing Science and Technology, vol. 1, no. 3, pp. 191–198, 2009.*
15. *J. A. Hurtado Alegra, M. C. Bastarrica, A. Quispe, and S. F. Ochoa, "MDE-based process tailoring strategy: MDE-BASED PROCESS TAILORING STRATEGY," Journal of Software: Evolution and Process, vol. 26, no. 4, pp. 386–403, Apr. 2014.*
16. *B. Curtis, Marc I Kellner, and, J Over, 1992. Process modeling D. L. Levin & F. C. Morriss, eds. Communications of the ACM, 35(9), pp.75-90.*
17. *OMG-SPEM2: Software & Systems Process Engineering Metamodel (SPEM) Version 2.0, http://www.omg.org/spec/SPEM/2.0/.*
18. *XPDL-Spec, www.omg.org/bpmn/.../XPDL_BPMN.pdf*
19. *R. Bendraou, M.P. Gervais, and X. Blanc, "UML4SPM: a UML 2.0-based metamodel for software process modeling". In: Briand, L., Williams, C. (eds.) MoDELS 2005. LNCS, vol. 3713, pp. 17-38. Springer, Montego Bay, Jamaica (2005*
20. *R. Elner, S. Al-Hilank, A. Bediaga, J. Drexler, M. Jung, D. Kips, and M. Philippssen, "eSPEM – A spem extension for enactable behavior modeling". In: Kuhne, T., Seloc, B., Gervais, M.P., Terrier, F. (eds.) ECMFA 2010. LNCS, vol. 6138, pp. 116-131. Springer, Paris (2010).*
21. *Bendraou R., Combemale B., Crégut X. Gervais., M.P.: Definition of an eXecutable SPEM2.0. In: 14th Asia-Pacific Software Engineering Conference (APSEC), pp. 390-397. IEEE Computer Society, Nagoya, Japan (2007).*
22. *Fall Ibrahima; Diaw Samba: A Metamodel for MDE Process Model-Products Relationships. 2016 IEEE 25th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE) Pages: 166 – 171*
23. *Kedji, K.A., Lbath, R., Coulette, B., Nassar, M., Baresse, L., Racaru, F., 2014. Supporting collaborative development using process models: a tooled integration-focused approach: SUPPORTING COLLABORATIVE DEVELOPMENT USING PROCESS MODELS. Journal of Software Evolution Process 26, 890–909. doi:10.1002/smr.1640*
24. *N. Koch., "Transformations techniques in the model-driven development process of UWE". In: 6th International Conference on Web Engineering (ICWE), Volume 155 Article N° 3. ACM, California (2006)*
25. *C. Kroiß, and N. Koch, "UWE metamodel and profile: user guide and reference". LMU, Technical Report (2008).*
26. *F. Jouault, F. Allilaire, J. Bézivin, I. Kurtev, and P. Valduriez, "ATL: a QVT-like transformation language," in Companion to the 21st ACM SIGPLAN symposium on Object-oriented programming systems, languages, and applications, 2006, pp. 719–720.*