An Y MDE Approach for Enactable Software Process Models Generation

Samba Diaw, Mamadou Lakhassane Cisse and Alassane Bah
Polytechnic Institute (ESP) @ Cheikh Anta Diop University (UCAD), UMMISCO Laboratory, Dakar, Senegal

Keywords: Model-driven Engineering (MDE), Model Transformations, MDE Software Process, Tailoring, Instantiation.

Abstract: The advent of MDE enabled to automate software development while reducing its production time. While software companies need continually to improve and customize their software processes, an automated 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 organisational and operational companies' needs. However, even if that tailored process can be used for a specific project, it still lacks resources needed for execution. In this short paper, we propose a Y model-based approach that allows tailoring software processes and generating enactable software process models by using models transformations. We defined metamodels to express models involved in those transformations. We illustrate our approach with an extract of the UWE Process which we adapt and instantiate for a development project with .Net.

1 INTRODUCTION

Nowadays, software applications become more and more important in our daily life. However, their implementation becomes more and more complex depending on their nature. To develop software with a high quality we should have a good 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. For a better management of a software development project, it is important to capture all changes that happen during the elaboration of the software meaning the description of the actual (i.e. real) process.

According to Curtis and 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" (Curtis and al., 1992). A process model is an abstract representation and does not capture concrete information on how the model-products are really managed during process execution. While software companies need continually to improve and customize their software processes, an automated 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. So, tailoring of process models to exactly fit organisational and operational companies' needs constitute a crucial task for the success of software development project. Tailoring could be a very difficult task, which typically has to take into consideration several human and organisational issues. In this respect, managing such a complexity with model-transformations is a challenging and ambitious issue.

To address this issue, we propose in this article, a Y model-based approach to tailor MDE processes and then generate enactable software process models. To validate our approach, we use an extract of the UWE process and apply it to a development project in .Net.

The remainder of the article is structured as follows: the section 2 presents the Y model-based approach while the section 3 presents the prototype. Section 4 deals with the validation of our Y approach with an illustrated example. The section 5 deals with related works. In the last section, we conclude this article and introduce some perspectives.

Diaw, S., Cisse, M. and Bah, A.
An Y MDE Approach for Enactable Software Process Models Generation.
DOI: 10.5220/0006649505110518
In *Proceedings of the 6th International Conference on Model-Driven Engineering and Software Development (MODELSWARD 2018)*, pages 511-518
ISBN: 978-989-758-283-7

2 THE Y MODEL-BASED APPROACH

Our contribution will consist in designing and implementing a Y model-based approach (Figure 1) to produce an enactable software process model from a generic process model.

The first step (tailoring) of this approach is to produce a tailored process model from a generic process model so-called PIPM (Project Independent Process Model). For the first step, the main idea is to consider tailoring as models' transformation that takes two input models:

- a generic software process model independent from any project (PIPM) conforms to SPEM4MDE,
- a model representing the context of a project/organization conforms to SPCM (Software Process Context Metamodel).

The second step is the instantiation of the tailored process model according to a project/organization context. As a result, an enactable process model so-called PSPM (Project Specific Process Model) will be produced and will include the actors/tools and tasks to be executed.

The Y model-based approach involves four metamodels:

- SPEM4MDE (Diaw and al., 2011): a metamodel taking into account the concepts of software development process definition,
- SPCM (Hurtado and al., 2011): a metamodel that defines the context model of a project/organization),
- PRM (Project Resources Metamodel) (Figure 2): a new metamodel that represents the resources of a given project
- EPM (Enactable Process Metamodel) (Figure 3): a new metamodel that defines managed elements at enactment time

Figure 2 describes the Project Resources metamodel. The resources used in a software development project are dependent to 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 used resources in the execution of a development project. This metamodel describes every resource element within a software development project.

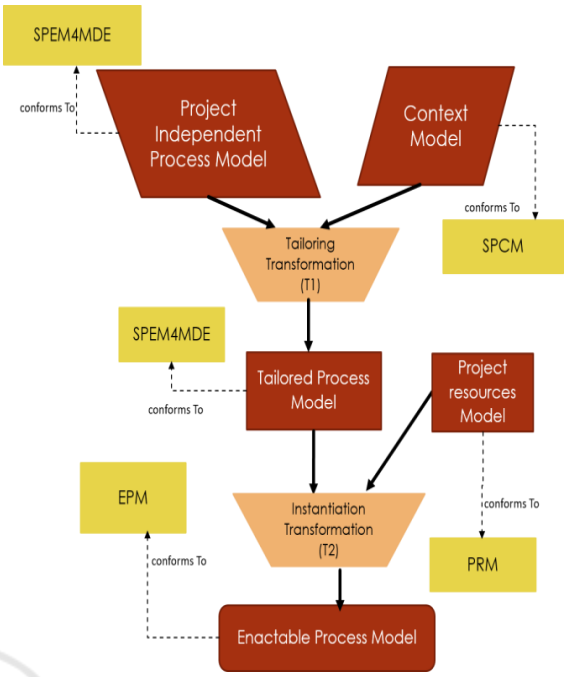

Figure 1: The Y model-based approach.

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 (primary or secondary). The resource type defines the type of resource, which can be human, software, hardware or any kind of resource type defined by the project manager. Resource types might not have the same properties. For solving that issue, we defined the ResourceProperty concept in order to define new properties for a resource type. Resource elements can be organized in groups. Only human resource elements are linked with their role. Examples of resource elements can be Bob (name), which is a human resource type. For the human resource type, we can also define different others properties such as 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.

Figure 3 presents the Enactable Process Metamodel (EPM). The instantiation process will produce a model ready to be enacted. The importance of this model is to give to the project manager all information about tasks and their performers. The model contains 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 resource that 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 the EPM metamodel. The main concepts of EPM are "TaskInstance", "MDEActivityInstance" and "TransformationInstance". They help defining the activities and tasks that are to be executed. Transformation instance is also a task instance. The task instance takes as parameters one or more model instances (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 ResourceElement concept from the Project Resources Metamodel to represent the specific resources (human, hardware, software) for executing a task or transformation.

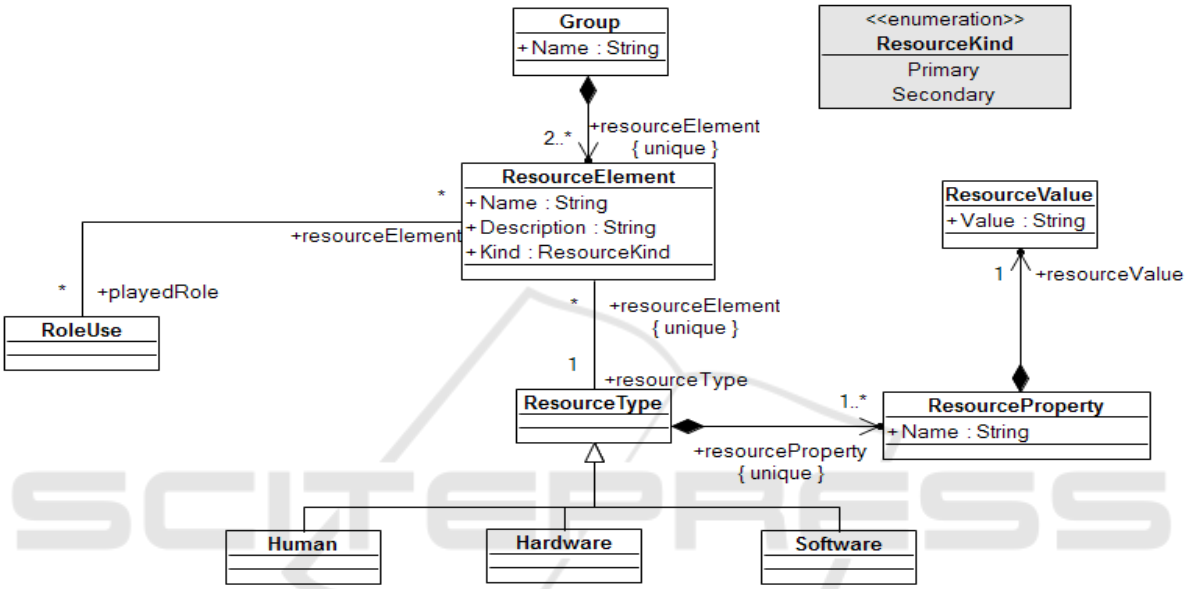

Figure 2: Project Resources Metamodel – PRM.

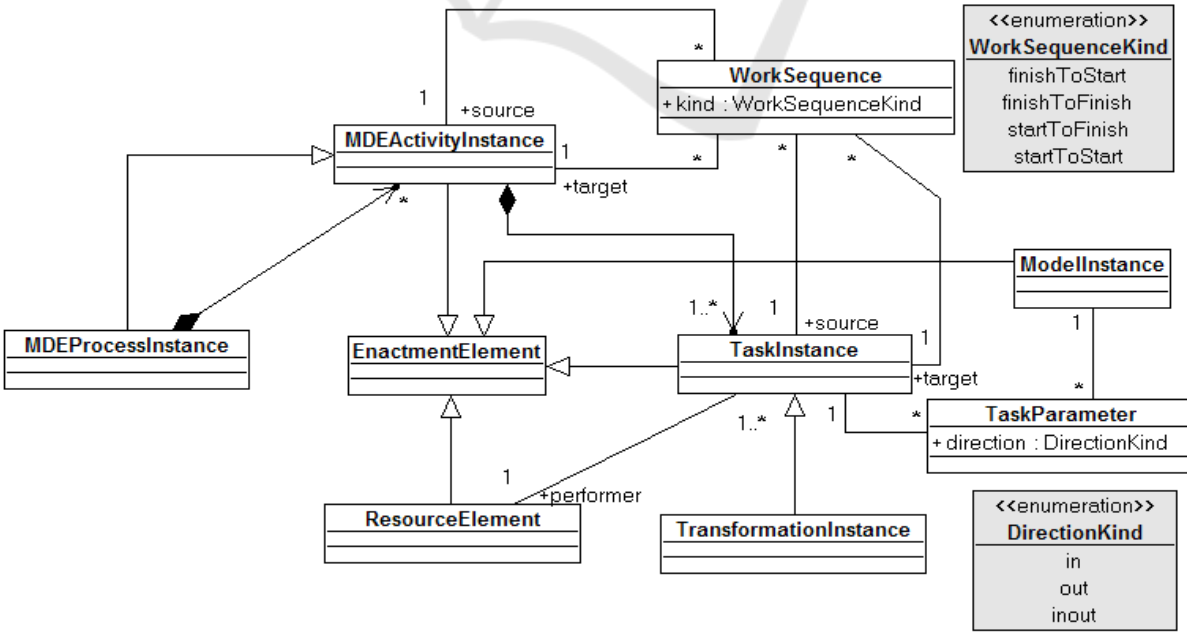

Figure 3: Enactable Process Metamodel.

3 THE PROTOTYPE

To validate our approach, we have developed a prototype. As shown by figure 4, the prototype is divided into two components: *SPEM4MDE Process Editor*, and *SPEM4MDE Process Enactment Engine*.

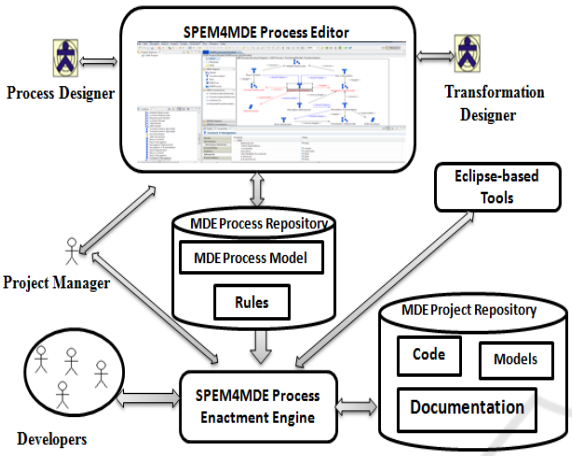

Figure 4: Architecture of SPEM4MDE-PSEE.

SPEM4MDE Process Editor allows process designers to describe, modify and tailor process models. Once the process model is described, the process designer may check it with respect to the constraints defined in the SPEM4MDE metamodel, or regarding to additional constraints. There are two ways for checking MDE process models: checking on demand (i.e. when the user triggers himself the checking process) or checking during edition (i.e. checking is done automatically by the tool). Outcomes of process editing are stored in a repository called *MDE Process Repository*. For instantiating a MDE process model in a given project, a project manager may also use this editing component.

SPEM4MDE Process Enactment Engine allows the project manager to instantiate a tailored process model and the developers to enact a project-specific process model by giving them their tasks and the current state of any process element. It is integrated with other eclipse-based tools (ATL, Smart QVT, Code Management Tool, etc.) in order to execute the activities of the instantiated process model. Developers can then keep track of what is the current state of each element of the MDE project, what has been done before and what is left. Outcomes (models, code, documentation, etc.) are stored in a MDE *Project Repository*

4 VALIDATION

To illustrate our Y model-based approach, we have chosen as an example the UWE (UML-based Web Engineering) process (Koch and al., 2006) (*Kroiß* and al., 2008). UWE is a process that covers web systems development cycle from requirements to code generation. Figure 5 represents an extract of the UWE process described with SPEM4MDE.

4.1 Project Independent Process Model (UWE Process)

After the description of the requirements model a first transformation (Req. 2 content) produces the content model. The UML standard may be used to describe the content model. The content model is used for the following activity (*Content 2 Navigation)* to produce the navigation model. From one content model, different navigation views can be obtained, e.g. for different stakeholders of the web system like anonymous user, registered user and administrator.

The requirements model contains information that is useful for the enrichment of the navigation model. For this purpose, the transformation (Req. 2 Navigation) is used.

The navigation model generated on the content model contains itself valuable information that allows for reasoning and improving the navigation model. For the transformation (Navigation Refinement) the following constrains are defined:

1. An index is added for all associations of the navigation model that have multiplicity greater than one at the directed association end.
2. All navigation classes that have at least one outgoing association require a menu class with menu items defined on basis of the association ends of the associations.

Presentation elements are generated based on navigation elements of the navigation model and merged then with style guide information.

For example, for each link in the navigation model an adequate anchor is required in the presentation model.

Functional models (content, navigation and presentation) are afterwards integrated mainly for the purpose of verification into a *big picture model*. Finally, the platform-specific code (Java, .Net) is generated from big picture *model*.

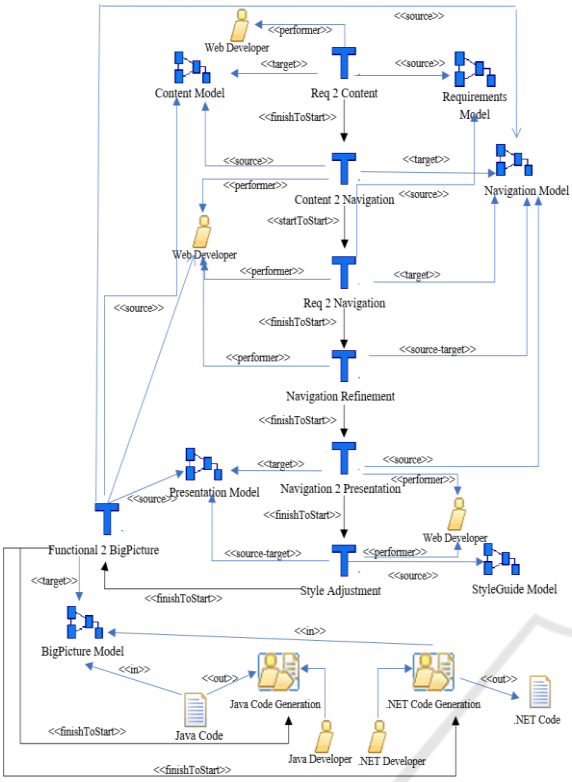

Figure 5: Generic UWE Process model.

4.2 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 model transformations. 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 *development platform*. SPCM allows us to create more variables but we rather stick to these two variables since they describe enough our context model.

Table 1: Context elements and values.

Context Attribute	***Value***
Project type	*Development Project*
Development Platform	*. NET*

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 are going to use ATL (Jouault and al., 2006) to define the tailoring transformation rules.

4.3 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 6 represents the resulting process after the tailoring activity. Only the required activities roles and artifacts are present. The resulting process does not include any additional activity. The "Req. 2 navigation" transformation is removed, as it is not mandatory when the navigation model produced by the "Content 2 Navigation" presentation is well defined. The "Java Code Generation" activity is also removed, as it is not mandatory for a .Net development project.

4.4 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 executed by MDE tools. The involved roles in the UWE process are:

- Web Developer
- Java Developer
- .Net Developer

To instantiate our tailored process model, we are going to choose real actors for those roles. For the role web developer, we can have a human resource with first name, last name, and address properties. Bob and Alice will then play the web developer role while Trevor will play the .Net developer.

4.5 The Resulting Enactable Process Model (Enactable UWE Process)

Once we have the tailored process, we can go through an instantiation activity 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 to execute them (Bob, Alice, Trevor). It still conforms to the EPM metamodel. Figure 7 shows the last step of our approach that produces the enactable process model.

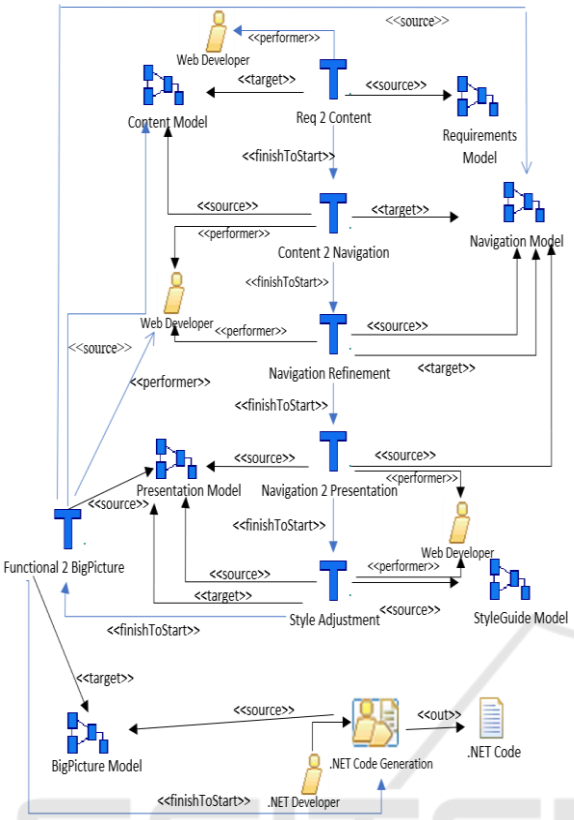

Figure 6: Tailored UWE process model.

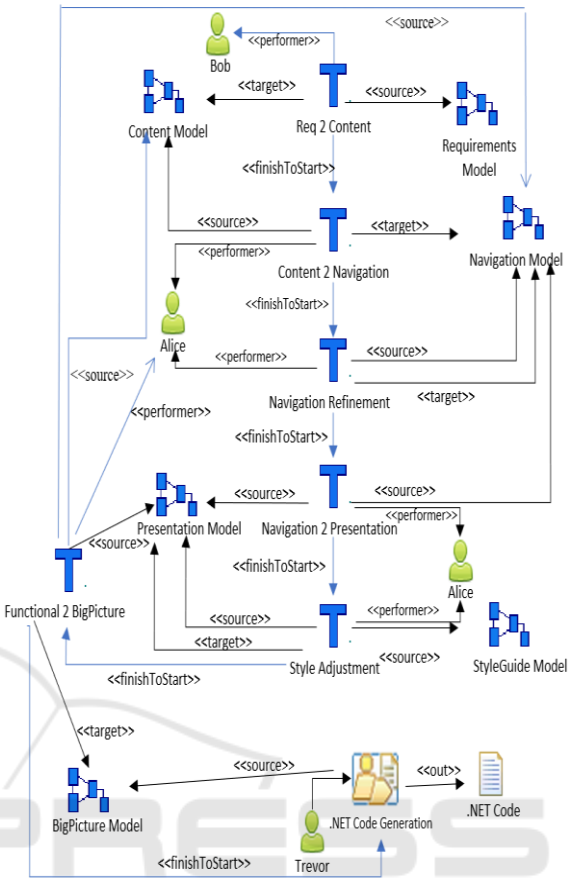

Figure 7: Enactable UWE Process Model.

5 RELATED WORKS

Process tailoring is the mechanism of adapting a software process to project needs (Silvestre and al., 2014).

In (Pedreira and al., 2017) different tailoring methods have been showed. 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 (Hanssen and al., 2005), 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 (Cao and al., 2004), a set of agile practices tailored for large-scale complex projects has been proposed.

In (González and al., 2014), an example of a 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 (Cockburn and al., 2004) 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 (Kalus and al., 2013) 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 (Xu and al., 2008), 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 (Martínez-Ruiz and al., 2012) 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 (Martínez-Ruiz and al., 2012). Figure 8 shows four major steps in software process tailoring.

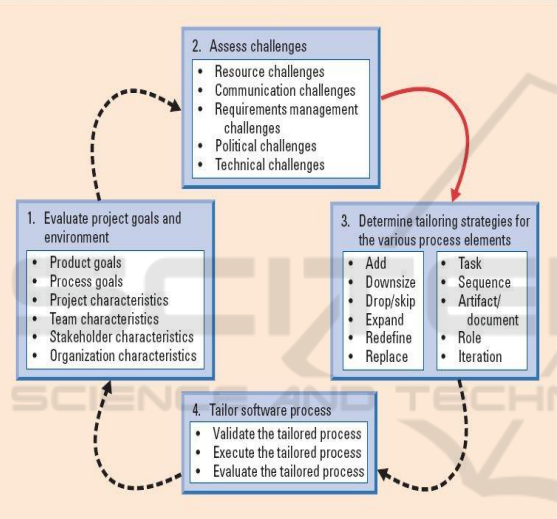

Figure 8: Software process tailoring steps(Martínez-Ruiz and al., 2012).

The criteria can also be split in four groups (Preez and al., 2009):

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

In (Hurtado and al., 2014), a model-based approach to software process tailoring has been proposed (figure 9). 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 process model still conforms to SPEM.

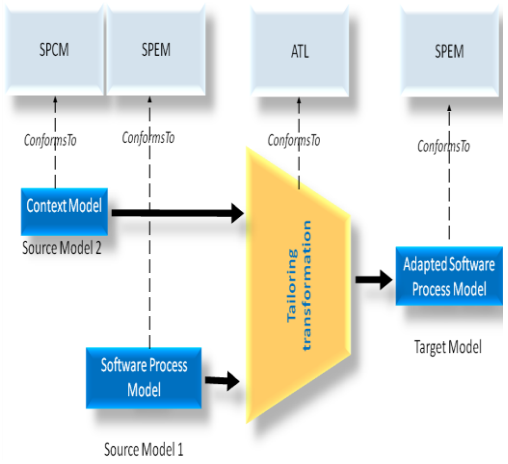

Figure 9: An MDE approach to tailoring (Hurtado and al., 20014).

5 CONCLUSION

In the literature, few approaches are natively supporting automatic process tailoring. Using MDE principles to tailor process model in a context of an organization or project and then generating an enactable process model is an ambitious issue.

To address this issue, we have presented a Y model-based approach that allows tailoring software processes and generating enactable software process models. Our approach involves two main activities tailoring and instantiation. The prototype we developed allows using an automated support to assist process designer in those two complex activities.

We validate our approach with an extract of the UWE process, which we adapt within a context of a .Net development project.

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

Cao L., Mohan K., Xu P., and Ramesh B., 2004. "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, pp. 10–pp*.

Cockburn A., 2004. Crystal clear: a human-powered methodology for small teams. *Pearson Education*.

Curtis B., Marc Kellner I, and, Over J., 1992. *Process modeling D. L. Levin & F. C. Morriss, eds. Communications of the ACM, 35(9), pp.75-90.*

Diaw S., Lbath R., and Coulette B., 2011. "Specification and Implementation of SPEM4mde, a metamodel for MDE software processes." in *SEKE, Miami - USA, pp. 646–653*.

Du Preez N., Lutters D., and Nieberding H., 2009. "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.*

Fall I., Diaw S?, 2016. A Metamodel for MDE Process Model-Products Relationships. *IEEE 25th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE) Pages: 166 – 171.*

González Pérez C and Henderson-Sellers B., 2008. Metamodeling for software engineering. *Chichester: Wiley*.

González F., Silvestre L., Solari M., and Bastarrica M. C., 2014. "Template Based vs. Automatic Process Tailoring," in *XXXIII International Conference of the Chilean Society of Computer Science (SCCC 2014)*.

Hanssen G. K., Westerheim H., and Bjørnson F. O., 2005. "Tailoring RUP to a defined project type: A case study," in *International Conference on Product Focused Software Process Improvement. Springer, pp. 314–327*.

Hurtado Alegra J. A., Bastarrica M. C., Quispe A., and Ochoa S. F., 2014. "MDE-based process tailoring strategy," *Journal of Software: Evolution and Process, vol. 26, no. 4, pp. 386–403, Apr*.

Hurtado Alegra J. A., Bastarrica M. C., Quispe A., and Ochoa S. F., 2011. "An MDE approach to software process tailoring," in *Proceedings of the 2011 International Conference on Software and Systems Process. ACM,* pp.43-pp52.(Online).Available:http://dl.acm.org/citation.cfm?id=1987885

Jouault F., Allilaire F., Bézivin J., Kurtev I., and Valduriez P., 2006. "ATL: a QVT-like transformation language," in Companion to the 21st ACM SIGPLAN symposium on Object-oriented programming systems, languages, and applications, pp. 719–720.

Kalus G. and Kuhrmann M., 2013. "Criteria for software process tailoring: a systematic review," in *Proceedings of the 2013 International Conference on Software and System Process. ACM, pp. 171–180*.

Koch N., 2006. "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*.

Kroiß C., and Koch N., 2008. "UWE metamodel and profile: user guide and reference". *LMU, Technical Report.*

Martínez-Ruiz T., Munch J., García F., and Piattini M., 2012. "Requirements and constructors for tailoring software processes: a systematic literature review," *Software Quality Journal, vol. 20, no. 1, pp. 229–260*.

Pedreira O., Piattini M., Luaces M. L., and Brisaboa M. R., 2007."A Systematic Review of Software Process Tailoring," vol. Volume 32 Issue 3. New York, NY, USA: *ACM SIGSOFT Software Engineering Notes, pp. 1–6*.

Silvestre L., Bastarrica M. C., and Ochoa S. F., 2014. "A model-based tool for generating software process model tailoring transformations," *in ModelDriven Engineering and Software Development (MODELSWARD), 2014 2nd International Conference on. IEEE, pp. 533–540*.

Xu P. and Ramesh, B., 2008. "Using process tailoring to manage software development challenges," *IT Professional, vol. 10, no. 4, pp. 39–45*.