DynaTool: A tool for optimizing hybrid software process

María Cecilia Bastarrica[1][0000−0002−8616−2144], Luis Silvestre[2][0000−0003−1806−8647], Andrés Wallberg, and Daniel González[1]

[1] Computer Science Department, University of Chile, Santiago, Chile
cecilia@dcc.uchile.cl, daniel.gonzalez.2@ug.uchile.cl
[2] Department of Computer Science - Faculty of Engineering, Curicó, Universidad de Talca, Chile
lsilvestre@utalca.cl

Abstract. Hybrid software processes that integrate agile and traditional practices are currently the most commonly used in the industry. Typically, development activities employ agile practices, while management tasks rely on more traditional methods. However, the optimal combination of practices depends not only on project attributes, such as team size, but also on the specific characteristics that need to be emphasized, such as time to market or early value addition. DynaTail was introduced as a method for integrating hybrid process tailoring with practice selection to optimize specific characteristics. While it received positive feedback in industry evaluations, users noted the need for a supporting tool to simplify the process for software developers, allowing them to focus on process elements rather than the technical details of the method. In response, we developed DynaTool, a model-driven engineering (MDE)-based support tool designed to meet this need. DynaTool offers interactive interfaces for specifying processes, context, and practices. In this paper, we refine DynaTail's optimization strategy to clearly identify the practices that should be applied to each process activity to maximize the desired characteristic. We also update DynaTool to incorporate these enhancements.

Keywords: Hybrid software process · MDE-based tool · software process evaluation.

1 Introduction

A software process is defined as a combination of roles, activities and work products [2,14]. Processes have been valued by software companies as a means for managing development in an organized manner so that it is possible to plan, schedule and provision software projects [23]. However, a single process does not fit all kinds of projects, even within the same organization. For example, the required developer experience depends largely on the complexity of the product, or the type of technologies that need to be used for its development [5,6]. Similarly, a small project could skip building several work products. The activity of adjusting the company's process to the particular characteristics of the project being addressed is called tailoring.

Agile methods propose a series of practices that software development teams adopt and adapt to address project development. These methods are specially appropriate to promote productivity in projects with high uncertainty but they do not provide strong support for project management [29]. Therefore, most companies follow hybrid processes, i.e., a combination of agile and traditional practices. Large companies used to define traditional software processes with strictly specified activities, responsibilities, workflow, work products, etc., but lately most of them start adopting some agile practices. Conversely, small companies that tried to follow a completely agile methodology, soon realized that some structure is required for managing projects under control [19]. However, it is not easy to assess which combination of agile and traditional practices adopted is the most appropriate one provided that this depends on the intended project characteristic as well as the project context [11,17].

The DynaTail method is designed for companies that must frequently adapt their hybrid processes to meet the diverse needs of their clients [31]. Tailoring, in this context, goes beyond simply adjusting the general process to fit a specific context; it also involves selecting different practices based on the characteristics that need optimization. DynaTail accomplishes this through two stages. First, the general process is tailored to align with the project's specific context. Second, it identifies the optimal combination of agile and traditional practices for executing the tailored process activities according to the desired characteristics.

DynaTail has been validated in a real software development company [21]. Although the company's process engineer highly appreciated the method, he highlighted the complexity of applying DynaTail without a supporting tool. We built DynaTool to support DynaTail based on the formal models defined in [31]. A first version of this tool was reported in [37]. Nevertheless, we argue that the complexity of the method, and not only the tool, could have also prevented the process engineer to fully grasp the potential and some of the features that could actually provide real hybrid optimized processes. In particular the first version did not explicitly defined the process along with the particular practices that should be applied for carrying out each activity.

We here present a refined version of DynaTail that automatically selects the set of practices for implementing the tailored process that optimizing the desired characteristic. We also present an updated implementation of DynaTool taking into account this improvement.

The rest of the paper is structured as follows. Section 2 presents background on software process improvement and tailoring, hybrid processes, and a general description of the DynaTail method. Section 3 describes DynaTool, its supporting models and transformations as well as the process for applying it [21] now using DynaTool. Finally, we describe our ongoing work and draw some conclusions in Section 4.

2 Background

2.1 Process Tailoring and Improvement

Software process improvement (SPI) is the area in software engineering that deals with the evaluation of software development processes, to assess and potentially improve them [13]. SPI used to follow strategies based on models such as CMMI and standards such as ISO that define a series of process areas that should be considered in order to reach a certain maturity level. This approach added repeatability but they have shown to be rigid for some kinds of applications, e.g. innovation projects.

Software process tailoring is the act of adjusting the activities of a process in order to create a new process suited to a different (and likely narrower) context [12]. But the context is not all that matters. Peng Xu and Balasubramaniam [39] and Vijayasarathy and Butler [36] found that tailoring is not only influenced by the project context, but also a set of environmental factors, challenges, project goals and process tailoring strategies.

Pedreira et al., in a systematic review [25], found that while most software process tailoring approaches involve some level of formality, the majority are only suitable for large organizations. Kalus and Kuhrmann [16] also conducted a systematic review on the criteria used for tailoring software processes. Their findings highlight the significance of specific organizational factors, which can serve as guidelines for selecting appropriate agile methods.

Software & Systems Process Engineering Metamodel (SPEM)[3] and Business Process Model & Notation (BPMN)[4] are the OMG standards for specifying software processes and business processes, respectively. Although SPEM is expressive for capturing subtleties of software processes, it lacks supporting tools. On the other hand, BPMN counts on a plethora of tools but it is not expressive for certain particularities of software development [9] such as modeling activities where several roles are involved or identifying variation points. However, in most cases BPMN supporting tools are enough.

SPEM and BPMN are the OMG standards for specifying software and business processes, respectively. While SPEM is well-suited for capturing the particularities of software processes, it lacks sufficient tool support. In contrast, BPMN benefits from a wide range of tools but falls short in expressing certain specific aspects of software development [9], such as modeling activities involving multiple roles [28] or identifying variation points. Nevertheless, in most cases, the tools available for BPMN are adequate. Pillat et al. [26] introduced BPMNt, an extension of BPMN designed to define variability in software processes modeled with BPMN. While this approach leverages BPMN's more user-friendly notation, deviating from the standard creates incompatibility with existing tools, hindering automatic transformations.

Hurtado et al. [15] propose a MDE-based strategy for software process tailoring. They consider a process specified in SPEM and a tailoring transformation that takes the process and the context models as input and yields a project specific process also specified in SPEM. This approach makes use of SPEM's variation primitives for identifying the process variation points.

Improving Agile processes involves optimizing how teams plan, execute, deliver, and gather feedback on development projects to streamline workflows and enhance efficiency [30]. Agile methodologies offer a framework for continuous improvement, empowering teams to respond to evolving business needs through the use of metrics, practices, and technology [4].

Campanelli et al. [3] conducted a systematic literature review on the tailoring of agile methods, including the various approaches used. Their analysis and classification revealed that most agile method tailoring techniques were independent of the specific

[3]http://www.omg.org/spec/SPEM/

[4]https://www.omg.org/bpmn/

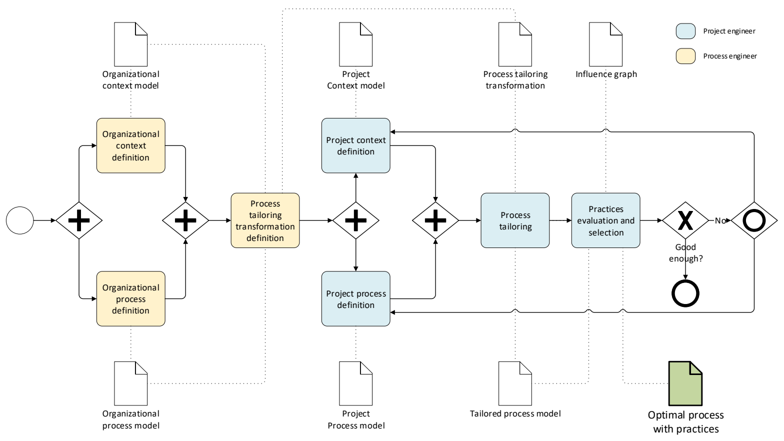

Project engineer
Process engineer
Organizational context model
Project Context model
Process tailoring transformation
Influence graph
Organizational context definition
Project context definition
Process tailoring transformation definition
Process tailoring
Practices evaluation and selection
No
Good enough?
Organizational process definition
Project process definition
Organizational process model
Project Process model
Tailored process model
Optimal process with practices

Fig. 1: DynaTail's process (extended from [37])

agile method employed by the organization. They also found that the tailored methods were primarily based on Scrum or XP, the tailoring approaches were grounded in method engineering (a meta-method process), and the tailoring criteria were largely driven by internal factors such as project type and communication.

Thiemich and Puhlmann [34] propose an integrated BPM project methodology framework that merges BPM with Scrum, focusing on the technical implementation of business processes. Martins and Zacarias [22] introduce an agile BPM methodology consisting of three key steps: process discovery, supervision, and assessment. They also compare various BPM methodologies, such as AGILIPO and Agile BPM Project. Zacarias et al.[40] compare four meta-models and propose an agile BPM meta-model. Von Bernardo et al.[1] link agility with BPM by presenting an agile BPM management method that spans four areas—analysis, planning, design, and building—where business, application, and technology goals are integrated into the analysis and planning phases.

Ozdenizci et al.[24] investigate the use of business process management methodologies to enhance agile software processes and agility maturity. Giachetti et al.[10] propose a model-driven approach for selecting agile practices, ensuring development processes align with quality standards.

2.2 Hybrid Software Process Improvement

Traditional software processes intend to bring structure into software development so that projects are easily managed. They define steps in order to avoid uncertainty and improvisation. However, in projects for innovative domains or not well defined requirements, these processes do not result effective. Agile software development methods have been proposed to deal with these difficulties.

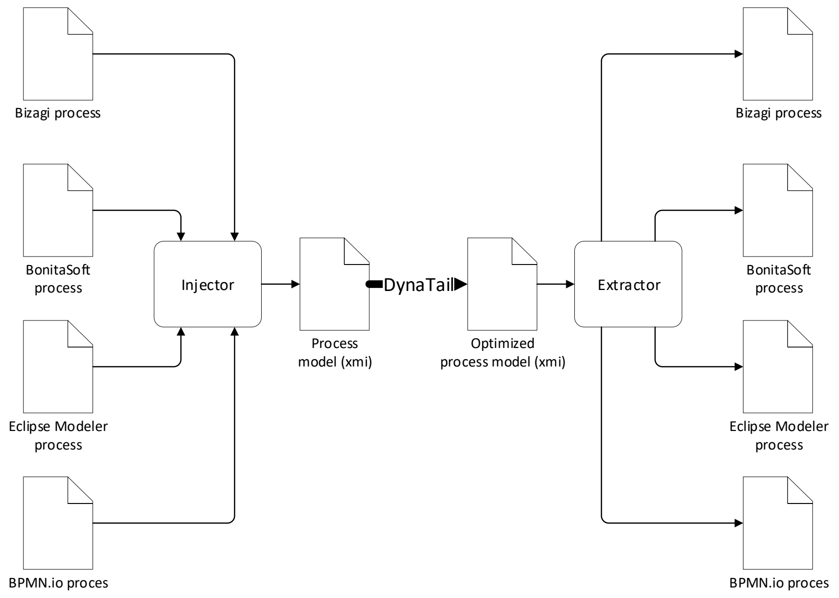


Fig. 2: DynaTool's architecture

Several companies have adopted agility, but completely agile projects are difficult to provision and schedule. Hybrid software processes that combine some agile and some traditional practices are a trade-off approach. Kuhrmann et al. [19] defined a hybrid software process as “any combination of agile and traditional approaches that an organizational unit adopts and customizes to its own context needs". One of the first proposal for hybrid software processes is “Water-Scrum-Fall" [38] where management activities are addressed with traditional practices while software development follows Scrum. But not any combination of practices is appropriate [27] for the organization and the project goals.

Evaluating each combination of practices presents a significant challenge [35]. Determining the appropriate level of agility is equally complex [7]: which activities should be handled by each approach, and which practices should be applied to each activity? While some empirical guidelines exist [33], both the desired characteristics of processes and the available practices evolve over time. Therefore, selecting the optimal combination of practices requires ongoing adjustment [17].

2.3 DynaTail

DynaTail acknowledges that context-based tailoring is not enough for process improvement, since two or more tailored processes may be consistent with the context but they improve different attributes. Moreover, the practices - either traditional or agile - that are applied for executing each activity, will also have an influence. Therefore, DynaTail explicitly considers these three dimensions: context, practices and desired attribute to

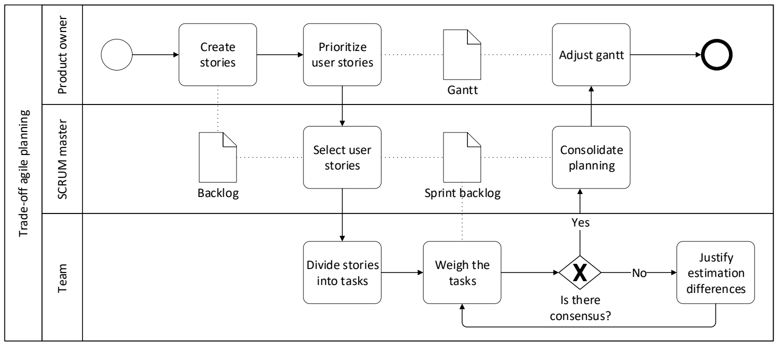


Fig. 3: Trade-off Agile Planning Process (from [37])

be improved for generating the appropriate process to be applied in a particular project with the purpose of optimizing certain particular characteristic.

The method involves two main activities: tailoring the process to the context, and choosing the appropriate practices that optimize the desired characteristic for implementing the activities of the tailored process. DynaTail defines tailoring as a model transformation that takes the process and context models and yields an adapted process model. The selection and evaluation of the most appropriate practices is carried out by computing the combination of practices in an influence graph that optimize the intended characteristic.

Here, the attributes that influence the intended characteristic to be improved are specified along with the weight of their influence. Similarly, the set of activities in the tailored process model may influence each of these attributes with different weights. Finally, each activity may be implemented with different practices, each one with its own influence weight. There is a different influence graph for each characteristic, and they are also specific to the organization since the included activities are those in the process model and the practices are the ones regularly applied in the organization.

The weights used in the evaluation are organization-specific, ranging from -2 to 2, as suggested by Diebold et al. [8]. A weight of -2 signifies a highly negative influence, while 2 indicates a highly positive influence. These weights are initially assigned by experienced developers within the organization and may be adjusted over time based on empirical results from previous projects. After evaluating the tailored process and its selected practices, the process engineers may determine whether the outcome is satisfactory. If not, they might choose to manually adjust the process or its context. Such modifications would also affect the selection of practices for implementing each activity. Activities and artifacts involved in DynaTail's process are identified in Figure 1.

3 DynaTool

In order to build a user-friendly supporting tool for DynaTail, the method has been fully formalized. An initial version of these DynaTail's models has already been presented in [31] where all activities and artifacts identified in Figure 1 are formalized.

DynaTool is based on these models and a first version was presented in [37]. In this paper we present a refinement of this tool that improves the evaluation activity in order to compute the optimal set of practices for implementing the process. This is represented in Figure 1 as “Practice evaluation and selection" updating what was originally activity called “Process evaluation". In this section, we replicate the process and context from the running example in the previous paper, providing a more detailed explanation of each element. Additionally, we introduce several new components: (1) an enhanced version of the tailoring rules definition interface, (2) an updated completely formalized influence graph model, and (3) the newly added “Practice Evaluation and Selection" activity, accompanied by two application scenarios.

3.1 Process Definition and Modeling

Organizations often establish specific processes to guide different types of projects. For instance, there might be distinct processes for system quality assurance, requirements specification, or project planning. These general processes should be tailored and adapted to fit the unique needs of each individual project. DynaTool uses BPMN for process formalization, provided that this notation is widely used in industry for process specification. Figure 3 illustrates the *Trade-off Agile Planning Process* process included in [21].

There are several modeling tools for specifying processes in BPMN such as BonitaSoft, Eclipse Modeler, BPMN.io and Bizagi. This has both benefits and drawbacks. Companies may choose any tool for modeling their processes but each of them implements its own *flavor* of BPMN adding some extra characteristics that are not necessarily compatible with BPMN 2.0 that is the strict standard.

To address this issue, we have built a set of projectors. First, an injector that transforms the BPMN process (BPMN file), that may be defined in any commercial BPMN modeling tool, into a BPMN 2.0 standard process model (XMI file). Second, an extractor that transforms the BPMN process model of the configured process (XMI file) back into a BPMN process description (BPMN file). This is necessary for visualizing in any BPMN modeling tool the result of applying DynaTool for obtaining the process to be applied in the particular project. Figure 2 describes DynaTool's architecture with respect to these projectors.

The projectors consider matching elements between the BPMN process and the BPMN process model. In this sense, there are process elements in BPMN processes that are not used for the injector such as those related to their layout, but that need to be preserved because they will be later needed for the extractor.

The injector applies the following steps: (1) Identify elements of the BPMN process that are relevant for building the process model, (2) Semantic analysis of the labels in the BPMN file, (3) Establish a dictionary that implements the matching elements, (4)

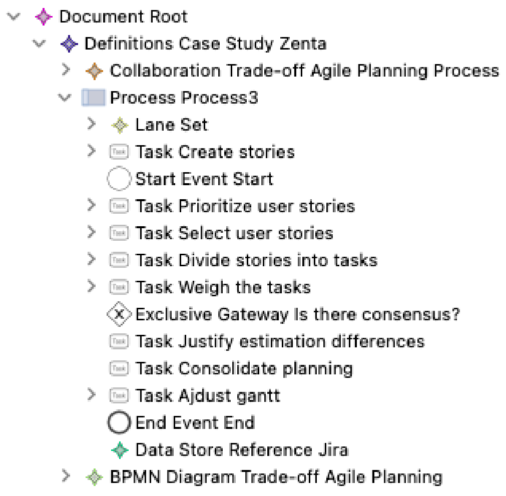

Document Root
Definitions Case Study Zenta
Collaboration Trade-off Agile Planning Process
Process Process3
Lane Set
Task Create stories
Start Event Start
Task Prioritize user stories
Task Select user stories
Task Divide stories into tasks
Task Weigh the tasks
Exclusive Gateway Is there consensus?
Task Justify estimation differences
Task Consolidate planning
Task Ajdust gantt
End Event End
Data Store Reference Jira
BPMN Diagram Trade-off Agile Planning

Fig. 4: Trade-off Agile Planning Process Model (from [37])

Create a hierarchical structure from the BPMN 2.0 metamodel, (5) Build the process model (XMI file).

Figure 3 shows the same Trade-off Agile Planning process presented in [37] defined using BonitaSoft while Figure 4 shows the BPMN process model generated after applying the injector. This BPMN process model conforms to the BPMN 2.0 metamodel and can be manipulated using EMF tools. However, this BPMN process model does not consider the graphical elements (only standard process elements).

The goal of DynaTail is to provide the process engineer or project manager the actual process that should be applied in a particular project including the details about the practices to be used for executing each activity. Therefore, this process should be displayed in a human understandable notation. To this end, the extractor that takes the process model resulting from applying DynaTail back to a process model that can be displayed in a BPMN modeling tool. The extractor applies the following steps: (1) Identify elements of the BPMN process model for building the BPMN process, (2) Semantic analysis of the labels in the XMI file, (3) Establish a dictionary that implements the matching elements, (4) Create a hierarchical structure from the BPMN description, (5) Build the BPMN process model (BPMN file). Provided that the XMI file does not contain information about process elements sequencing, the original BPMN file is also used as input in order in order to make use of the elements' graphical layout to build the BPMN file for displaying the process.

3.2 Influence graph definition

Each organization counts on an *Influence graph* for each potential characteristic that may need to be optimized. Here, the set of practices - either agile or traditional - that may implement each activity in the process are specified. Also, this graph specifies

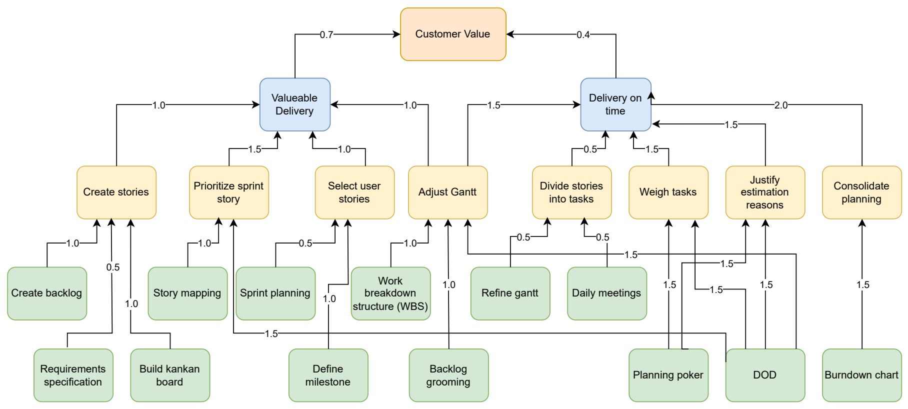


Fig. 5: Influence Graph

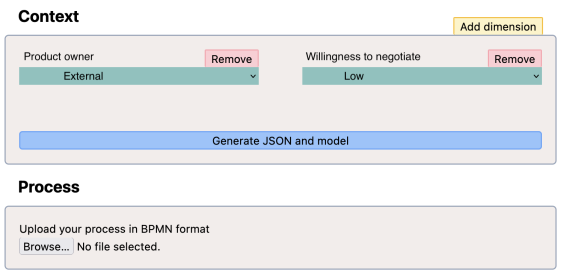
how much each of these practices influences the corresponding activity. Provided that BPMN does not count on variability primitives, we specify the process variability in this influence graph.

Figure 10 shows the *Influence graph model* for the process in Figure 3. The graph is structured in four levels. First, the characteristic to be optimized that in our example is “Customer value". A second level of attributes that, according to the literature, have an influence on that characteristic. In our case, these are “Valuable delivery" and “Delivery on time". Third, we have the set of activities on the process model that influence each attribute. And finally, the set of agile practices that may be used for implementing each of these activities. Each of these influences is quantified with a value between -2 and 2 that is initially defined by experts that usually apply the process and then they are adjusted according to empirical results. This graph conformes to an Influence graph metamodel that refines that presented in [37] for including practices explicitly relating them to the activities they may implement.

In order to better illustrate the way this *Influence graph* is used to obtain the optimized process, we have significatively extended the graph by adding several alternative practices that may be used for implementing each activity. Each of these practices has a different influence weight on the activity. For example, “Valuable delivery" has an influence of 0.7 on “Customer value", while “Delivery on time" has an 0.4 influence. Relating activities, the “Adjust Gantt" activity may implemented with three different practices, each one with its own weight: “DOD" 1.5, “Backlog grooming" 1.0, and “Work breakdown structure" 1.0.

3.3 Context Definition and Modeling

A project’s context is defined by the characteristics of its environment. Common characteristics in software development projects include system size, software complexity, development team size, and domain knowledge, among others. While a company may establish a standardized development process and apply it consistently, it can also take


Context

Product owner: External

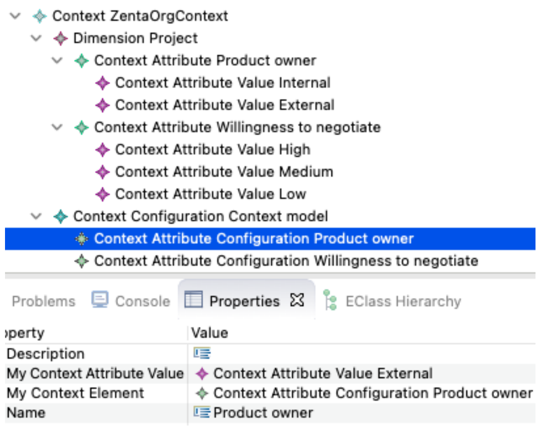
Willingness to negotiate: Low

Process

Upload your process in BPMN format

No file selected.

Fig. 6: Context definition tool user interface (from [37])


```
Context ZentaOrgContext
  Dimension Project
    Context Attribute Product owner
      Context Attribute Value Internal
      Context Attribute Value External
    Context Attribute Willingness to negotiate
      Context Attribute Value High
      Context Attribute Value Medium
      Context Attribute Value Low
  Context Configuration Context model
    Context Attribute Configuration Product owner
    Context Attribute Configuration Willingness to negotiate
```

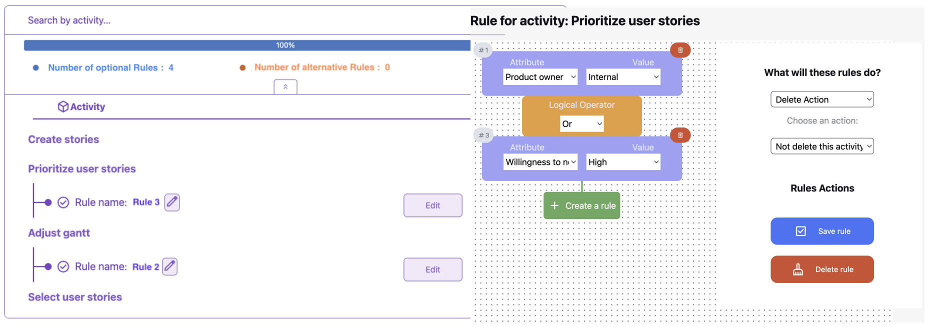
perty	Value
Description	
My Context Attribute Value	Context Attribute Value External
My Context Element	Context Attribute Configuration Product owner
Name	Product owner

Fig. 7: Context model (from [37])

contextual characteristics into account to select or tailor the process, making it more suitable for each specific project. Although any characteristic could theoretically be used to define the context for process tailoring, research has shown that project/product size and application domain are the most commonly considered dimensions [18]. In this paper, we adopt a broad perspective and consider any type of dimension.

To develop DynaTool, we formally represent the project context as a model that integrates seamlessly into the tool. We have created a custom context definition interface that enables the process engineer to specify the context attributes to be considered, while the project engineer can assign values to these attributes for a specific project. Additionally, we have developed an injector that takes the context, defined as a JSON file generated by this tool, and converts it into a context model.

In order to replicate the running example presented in [37], we show in Figure 1 the same context. Figure 6 shows the interface through which the project manager defines the context, while Figure 7 illustrates the resulting context model after applying the injector. The context model comprises two sections: the *Organizational Context Model*, which defines all context attributes and their potential values, and the *Project Context Model*, which configures these attributes with specific values for a given project con-


Search by activity...

100%

Number of optional Rules : 4

Number of alternative Rules : 0

Activity

Create stories

Prioritize user stories

Rule name: Rule 3

Adjust gantt

Rule name: Rule 2

Select user stories

Rule for activity: Prioritize user stories

Attribute: Product owner — Value: Internal

Logical Operator: Or

Attribute: Willingness to n — Value: High

What will these rules do? Delete Action

Choose an action: Not delete this activity

(a) Selecting variable activity (b) Defining transformation rules

Fig. 8: Transformation rules user interface

text. Only attributes defined in the Organizational Context Model can be configured. As indicated in the interface, the *Project Owner* attribute in the context model is set to “External", as specified at the bottom part of the model. Similarly, the *Willingness to Negotiate* attribute is set to “Low".

In the lower part of the context definition interface there is the possibility of uploading the BPMN process file and lounge first the injector and then the tailoring transformation described in the following section.

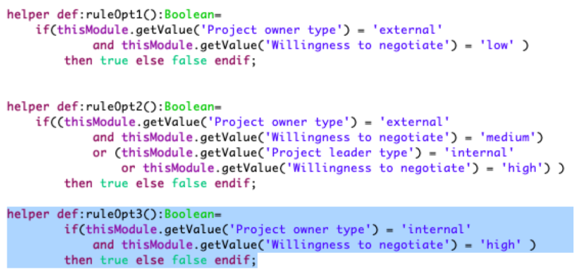
3.4 Process Tailoring Definition and Modeling

We developed a new interactive tool for defining transformation rules for process model tailoring. The generated transformations follow the same syntax as before. Using a separate model injector, these rules are then transformed into an ATL transformation model, which takes both the process and context models as input and produces a tailored process model as output. In what follows we describe each of these activities, their user interfaces and their supporting models.

Writing tailoring rules in a formal language is highly complex and could pose a significant barrier to making DynaTool applicable across various processes and contexts in industry. To address this, we designed a even more user-friendly interface that allows process engineers to define these tailoring rules interactively.

To this end, the *Process model* and *Context model* must be defined in advance. First, DynaTool retrieves all activities from the Process model (see Figure 8a). The process engineer can then select an activity that may be implemented differently based on specific context values. A rule will define for example that in certain contexts, the activity should not be necessary to be performed as part of the optimized process.

For instance, in this example, “Prioritize user stories" has been selected. A specific rule for this activity can then be defined using the user interface shown in Figure 8b. Figure 9 presents three rules defined by the process engineer for this activity. For example, rule 3 is the one generated according to the specification in Figure 8: *Product owner* is classified as “internal" or their *Willingness to negotiate* is “high," as outlined in the



```
helper def:ruleOpt1():Boolean=
    if(thisModule.getValue('Project owner type') = 'external'
            and thisModule.getValue('Willingness to negotiate') = 'low' )
        then true else false endif;


helper def:ruleOpt2():Boolean=
    if((thisModule.getValue('Project owner type') = 'external'
            and thisModule.getValue('Willingness to negotiate') = 'medium')
            or (thisModule.getValue('Project leader type') = 'internal'
                or thisModule.getValue('Willingness to negotiate') = 'high') )
        then true else false endif;

helper def:ruleOpt3():Boolean=
        if(thisModule.getValue('Project owner type') = 'internal'
            and thisModule.getValue('Willingness to negotiate') = 'high' )
        then true else false endif;
```

Fig. 9: Tailoring rules automatically generated (adapted from [37]

context specification of the previous section, then the *Prioritize stories* activity must be included in the tailored process. In this case, if no other rule applies, the process will remain as shown in Figure 3, with no activities removed. Similarly, the first rule is the one that would be applied in the context defined in Figure 6.

Once all the desired rules have been defined, the transformation shown in Figure 9 can be automatically generated. It is important to note that this operation is a higher-order transformation, which is highly complex and challenging to perform manually.

3.5 Practice evaluation and selection

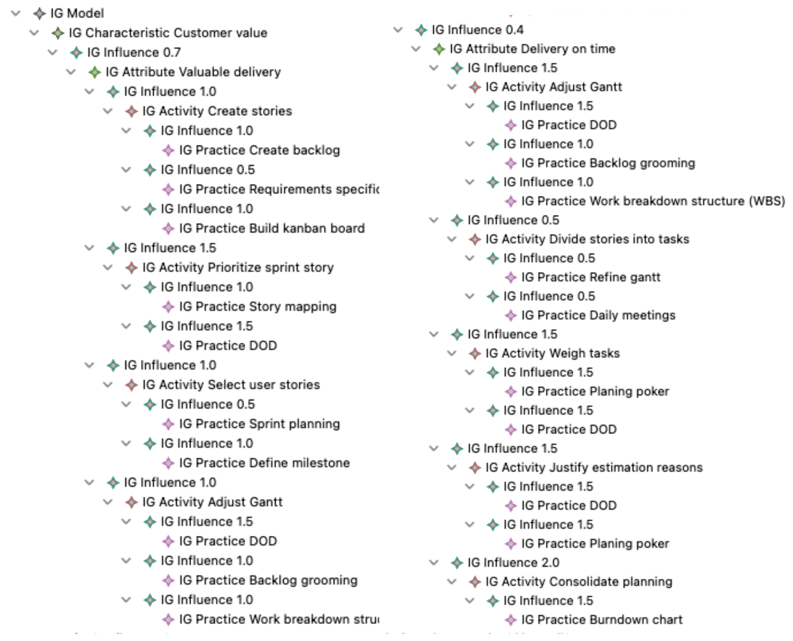
The *Practice evaluation and selection* activity extends the *Process evaluation* activity introduced in the first version of DynaTool [37]. This step is implemented as a model transformation that takes the *Influence graph* and the *Tailored process model* as inputs, producing a process optimized with a specific set of practices aimed at enhancing the desired characteristic. In this step, the practices that most effectively contribute to optimizing the targeted characteristic are selected. It is important to note that only the activities included in the tailored process will be evaluated and considered for the final output.

The characteristic can either be maximized or minimized. If the goal is to maximize it, as in the case of adding value to the client, the practices selected should have the greatest positive influence on the relevant activities. Conversely, if the characteristic needs to be minimized, such as reducing time to market, the chosen practices should be those with the least influence on their corresponding activities.

Tables 1 and 2 show the influence of all the practices on the process activities. This information is the same included in the *Influence graph* depicted in Figure 10.

In order to maximize the desired characteristic (value added in the example), the value of the attributes that influence it directly must be also maximized. Provided that the activities are those that are part of the process, the practice chosen for implementing each activity will be that that also maximizes the influence of the activity over the attribute.

The value of the attributes is computed as the average influence of all the influence of the activities along with their chosen practices as shown in Table 3 and Table 4 (we consider two possible evaluations).



```
IG Model
  IG Characteristic Customer value
    IG Influence 0.7
      IG Attribute Valuable delivery
        IG Influence 1.0
          IG Activity Create stories
            IG Influence 1.0
              IG Practice Create backlog
            IG Influence 0.5
              IG Practice Requirements specifi
            IG Influence 1.0
              IG Practice Build kanban board
        IG Influence 1.5
          IG Activity Prioritize sprint story
            IG Influence 1.0
              IG Practice Story mapping
            IG Influence 1.5
              IG Practice DOD
        IG Influence 1.0
          IG Activity Select user stories
            IG Influence 0.5
              IG Practice Sprint planning
            IG Influence 1.0
              IG Practice Define milestone
        IG Influence 1.0
          IG Activity Adjust Gantt
            IG Influence 1.5
              IG Practice DOD
            IG Influence 1.0
              IG Practice Backlog grooming
            IG Influence 1.0
              IG Practice Work breakdown stru
```

```
IG Influence 0.4
  IG Attribute Delivery on time
    IG Influence 1.5
      IG Activity Adjust Gantt
        IG Influence 1.5
          IG Practice DOD
        IG Influence 1.0
          IG Practice Backlog grooming
        IG Influence 1.0
          IG Practice Work breakdown structure (WBS)
    IG Influence 0.5
      IG Activity Divide stories into tasks
        IG Influence 0.5
          IG Practice Refine gantt
        IG Influence 0.5
          IG Practice Daily meetings
    IG Influence 1.5
      IG Activity Weigh tasks
        IG Influence 1.5
          IG Practice Planing poker
        IG Influence 1.5
          IG Practice DOD
    IG Influence 1.5
      IG Activity Justify estimation reasons
        IG Influence 1.5
          IG Practice DOD
        IG Influence 1.5
          IG Practice Planing poker
    IG Influence 2.0
      IG Activity Consolidate planning
        IG Influence 1.5
          IG Practice Burndown chart
```

(a) Customer Value Characteristic - Valuable delivery Attribute

(b) Customer Value Characteristic - Delivery on time Attribute

Fig. 10: Influence Model

Table 1: I(Prac,Act): Matrix of influences of practices on activities

Activity / **Practice**	Create stories	Prioritize sprint story	Select user stories	Adjust gantt
Create backlog	1.0	0	0	0
Requirements specification	0.5	0	0	0
Build kanban board	1,0	0	0	0
Story mapping	0	1.0	0	0
DOD	0	1.5	0	1.5
Sprint planning	0	0	0.5	0
Define milestone	0	0	1.0	0
Backlog grooming	0	0	0	1.0
Work breakdown structure (WBS)	0	0	0	1.0
Refine gantt	0	0	0	0
Daily meetings	0	0	0	0
Planing poker	0	0	0	0
Burndown chart	0	0	0	0

Finally, the characteristic value is obtained as the average of the value of these attributes. In this case is 1.35. If the value obtained is considered good enough, the method

Table 2: I(Prac,Act): Matrix of influences of practices on activities

Practice \ **Activity**	Divide stories into tasks	Weigh the tasks	Justify estimation reasons	Consolidate planning
Create backlog	0	0	0	0
Requirements specification	0	0	0	0
Build kanban board	0	0	0	0
Story mapping	0	0	0	0
DOD	0	1.5	1.5	0
Sprint planning	0	0	0	0
Define milestone	0	0	0	0
Backlog grooming	0	0	0	0
Work breakdown structure (WBS)	0	0	0	0
Refine gantt	0.5	0	0	0
Daily meetings	0.5	0	0	0
Planing poker	0	1.5	1.5	0
Burndown chart	0	0	0	1.5

Table 3: Influence of activities and practices on attributes - First evaluation -Agile practices

Activity	**Practice**	**Valuable delivery**	**Delivery on time**
Create stories	Build kanban board	1.0 * 1.0	0
Prioritize sprint story	DOD	1.5 * 1.5	0
Select user stories	Define milestone	1.0 * 1.0	0
Adjust gantt	DOD	1.5 * 1.0	1.5 * 1.5
Divide stories into tasks	Daily meetings	0	0.5 * 0.5
Weigh the tasks	Planning poker	0	1.5 * 1.5
Justify estimation reasons	Planning poker	0	1.5 * 1.5
Consolidate planning	Burndown chart	0	1.5 * 2.0
		0.84	1.25

ends and the best process for optimizing the intended characteristic is the "Hybrid process model" that defines not only the activities to be followed but also the practices that should be applied in each step.

On the contrary, if the process engineer considers that the resulting value is not good enough, he/she may proceed to *Change process* or *Change context* and restart whole method. Modifying the process could be for example, adding new steps or roles not present in the previous process, while modifying the context could be for example, adding more developers to the team in charge of the project. Moreover, another potential change could be adding new practices to the influence graph that can potentially be applied for performing certain activities.

Finally, we apply the extractor that takes as input the XMI tailored process and generates the BPMN process. The BPMN process is automatically generated and can be visualized from a BPMN tool as Bizagi, BonitaSoft, BPMN.io or Eclipse Process Modeler.

Table 4: Influence of activities and practices on attributes - Second evaluation - Hybrid practices

Activity	Practice	Valuable delivery	Delivery on time
Create stories	Requirements specification	0.5 * 1.0	0
Prioritize sprint story	DOD	1.5 * 1.5	0
Select user stories	Define milestone	1.0 * 1.0	0
Adjust gantt	WBS	1.0 * 1.0	1.0 * 1.5
Divide stories into tasks	Refine gantt	0	0.5 * 0.5
Weigh the tasks	DOD	0	1.5 * 1.5
Justify estimation reasons	DOD	0	1.5 * 1.5
Consolidate planning	Burndown chart	0	1.5 * 2.0
		0.59	1.17

Since the primary goal of DynaTail is to define the optimal setup for a given project encompassing the process, context, and practices that best achieve the desired characteristic—the notion of “good enough" is largely subjective and depends on the process engineer’s judgment. However, a more objective approach can be taken using a *what-if* strategy, which allows for comparing different configurations. Ultimately, it is still up to the process engineer to decide, for example, whether the organization has the resources to configure a specific context or implement certain practices.

4 Conclusions

A software process is considered effective not only if it aligns with its context but also if it helps achieve a desired goal, such as minimizing development time or maximizing value added. While agile software development encourages the adoption and adaptation of various practices, it is not immediately clear which combination of practices will be most suitable for reaching the project’s specific objectives.

We have developed DynaTail, a strategy for determining the optimal configuration of agile and traditional practices to achieve a desired value for a specific characteristic. To facilitate the application of DynaTail, we created DynaTool, a model-driven engineering (MDE)-based support tool. This tool was initially introduced in [37].

In this work, we have enhanced the metamodels and refined the tool’s calculation methods to make it more powerful and user-friendly. Additionally, and most importantly, the updated version provides project managers with precise guidance on which practices should be applied to each activity—a feature that was less clear in the previous version.

Although the new version of the tool has not yet been fully applied in industry, we have successfully replicated all the processes from our previous work [20,21,32,37].

References

1. Bernardo Junior, R., de Padua, S.I.D.: Toward agile business process management: Description of concepts and a proposed definition. Knowledge and process management **30**(1), 14–32 (2023)

2. Braude, E.J., Bernstein, M.E.: Software engineering: modern approaches. Waveland Press (2016)
3. Campanelli, A.S., Parreiras, F.S.: Agile methods tailoring a systematic literature review. J. Syst. Softw. **110**(C), 85–100 (dec 2015). `https://doi.org/10.1016/j.jss.2015.08.035`, `https://doi.org/10.1016/j.jss.2015.08.035`
4. Choraś, M., Springer, T., Kozik, R., Lopez, L., Martínez-Fernández, S., Ram, P., Rodriguez, P., Franch, X.: Measuring and improving agile processes in a small-size software development company. IEEE access **8**, 78452–78466 (2020)
5. Clarke, P., O'Connor, R.V.: The situational factors that affect the software development process: Towards a comprehensive reference framework. Information and Software Technology **54**(5), 433–447 (2012)
6. Dayyala, N., Walstrom, K.A., Bagchi, K.K., Udo, G.: Factors impacting defect density in software development projects. International Journal of Information Technologies and Systems Approach (IJITSA) **15**(1), 1–23 (2022)
7. Diebold, P., Zeher, T.: The Right Degree of Agility in Rich Processes. In: Managing Software Process Evolution, pp. 15–37. Springer (2016)
8. Diebold, P., Zehler, T.: The agile practices impact model: idea, concept, and application scenario. In: Proceedings of the 2015 International Conference on Software and System Process. pp. 92–96. ACM (2015)
9. Dumas, M., Pfahl, D.: Modeling Software Processes Using BPMN: When and When Not? In: Managing Software Process Evolution: Traditional, Agile and Beyond – How to Handle Process Change, pp. 165–183. Springer International Publishing (2016)
10. Giachetti, G., de la Vara, J.L., Marín, B.: A model-driven approach to adopt good practices for agile process configuration and certification. Computer Standards & Interfaces **86**, 103737 (2023)
11. Gill, A.Q., Henderson-Sellers, B., Niazi, M.: Scaling for agility: A reference model for hybrid traditional-agile software development methodologies. Information Systems Frontiers **20**, 315–341 (2018)
12. Ginsberg, M.P., Quinn, L.H.: Process tailoring and the software capability maturity model. Citeseer (1995)
13. Humphrey, W.: A Discipline for Software Engineering. SEI Series in Software Engineering, Addison Wesley (1995)
14. Humphrey, W.S.: The software engineering process: definition and scope. In: Proceedings of the 4th International Software Process Workshop on Representing and Enacting the Software Process. pp. 82–83 (1988)
15. Hurtado Alegría, J.A., Bastarrica, M.C., Ochoa, S.F., Simmonds, J.: MDE software process lines in small companies. J. Syst. Softw. **86**(5), 1153–1171 (2013)
16. Kalus, G., Kuhrmann, M.: Criteria for software process tailoring: a systematic review. In: International Conference on Software and System Process. pp. 171–180. ACM (2013)
17. Klünder, J., Hebig, R., Tell, P., Kuhrmann, M., Nakatumba-Nabende, J., Heldal, R., Krusche, S., Fazal-Baqaie, M., Felderer, M., Bocco, M.F.G., et al.: Catching up with method and process practice: An industry-informed baseline for researchers. In: 2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP). pp. 255–264. IEEE, IEEE / ACM (2019)
18. Klünder, J., Karajic, D., Tell, P., Karras, O., Münkel, C., Münch, J., MacDonell, S.G., Hebig, R., Kuhrmann, M.: Determining context factors for hybrid development methods with trained models. In: International Conference on Software and System Processes, ICSSP'2020. pp. 61–70. ACM (2020)
19. Kuhrmann, M., Diebold, P., Münch, J., Tell, P., Garousi, V., Felderer, M., Trektere, K., McCaffery, F., Linssen, O., Hanser, E., Prause, C.R.: Hybrid software and system development

in practice: Waterfall, scrum, and beyond. In: Proceedings of the 2017 International Conference on Software and System Process. p. 30–39. ICSSP 2017, Association for Computing Machinery, New York, NY, USA (2017)
20. Marín, J., Bastarrica, M.C., Hurtado, J.A., Silvestre, L.: Dynatail: A method for hybrid software process tailoring. Tech. Rep. TR/DCC-2021-1, Computer Science Department, University of Chile (2021), https://www.dcc.uchile.cl/reportes
21. Marín, J., Hurtado, J.A., Bastarrica, M.C., Silvestre, L.: Tailoring hybrid software processes in a medium-size software company. In: Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing, SAC. pp. 1042–1050. ACM (2023)
22. Martins, P.V., Zacarias, M.: An agile business process improvement methodology. Procedia Computer Science **121**, 129–136 (2017)
23. Münch, J., Armbrust, O., Kowalcyzk, M., Soto, M.: Software Process Definition and Management. Springer-Verlag, Germany (2012)
24. Ozdenizci Kose, B.: Business process management approach for improving agile software process and agile maturity. Journal of software: Evolution and Process **33**(4), e2331 (2021)
25. Pedreira, O., Piattini, M., Luaces, M.R., Brisaboa, N.R.: A systematic review of software process tailoring. SIGSOFT Softw. Eng. Notes **32**(3), 1–6 (2007)
26. Pillat, R.M., Oliveira, T.C., Alencar, P.S.C., Cowan, D.D.: BPMNt: A BPMN extension for specifying software process tailoring. Inf. Softw. Technol. **57**, 95–115 (2015). `https://doi.org/10.1016/j.infsof.2014.09.004`, `https://doi.org/10.1016/j.infsof.2014.09.004`
27. Prenner, N., Unger-Windeler, C., Schneider, K.: Goals and challenges in hybrid software development approaches. Journal of Software: Evolution and Process **33**(11), e2382 (2021)
28. Pulgar, J., Bastarrica, M.C.: Transforming multi-role activities in software processes into business processes. In: Business Process Management Workshops. Revised Papers. vol. 281, pp. 372–383. Springer (2016)
29. Raharjo, T., Purwandari, B.: Agile Project Management Challenges and Mapping Solutions: A Systematic Literature Review. In: Proceedings of the 3rd International Conference on Software Engineering and Information Management. p. 123–129. ACM (2020)
30. Santana, C., Queiroz, F., Vasconcelos, A., Gusmão, C.: Software process improvement in agile software development a systematic literature review. In: 2015 41st Euromicro Conference on Software Engineering and Advanced Applications. pp. 325–332. IEEE (2015)
31. Silvestre, L., Bastarrica, M.C., Hurtado, J.A., Marín, J.: Formalizing the Goal-directed and Context-based Software Process Tailoring Method. In: XLVII Latin American Computing Conference, CLEI. pp. 1–9 (2021)
32. Silvestre, L., Bastarrica, M.C., Ochoa, S.F.: A model-based tool for generating software process model tailoring transformations. In: 2014 2nd International Conference on Model-Driven Engineering and Software Development (MODELSWARD). pp. 533–540. IEEE (2014)
33. Tell, P., Klünder, J., Küpper, S., Raffo, D., MacDonell, S.G., Münch, J., Pfahl, D., Linssen, O., Kuhrmann, M.: What are hybrid development methods made of?: an evidence-based characterization. In: Proceedings of the International Conference on Software and System Processes, ICSSP 2019, Montreal, QC, Canada, May 25-26, 2019. pp. 105–114. IEEE / ACM (2019)
34. Thiemich, C., Puhlmann, F.: An agile BPM project methodology. In: Business Process Management: 11th International Conference, BPM 2013, Beijing, China, August 26-30, 2013. Proceedings. pp. 291–306. Springer (2013)
35. Unterkalmsteiner, M., Gorschek, T., Islam, A.M., Cheng, C.K., Permadi, R.B., Feldt, R.: Evaluation and measurement of software process improvement—a systematic literature review. IEEE Transactions on Software Engineering **38**(2), 398–424 (2012). `https://doi.org/10.1109/TSE.2011.26`

36. Vijayasarathy, L.R., Butler, C.W.: Choice of Software Development Methodologies: Do Organizational, Project, and Team Characteristics Matter? IEEE Software **33**(5), 86–94 (2016)
37. Wallberg, A., González, D., Silvestre, L., Bastarrica, M.C.: A Tool for Modeling and Tailoring Hybrid Software Processes. In: Proceedings of the 12th International Conference on Model-Based Software and Systems Engineering, MODELSWARD 2024, Rome, Italy, February 21-23, 2024. pp. 264–271. SCITEPRESS (2024)
38. West, D., Gilpin, M., Grant, T., Anderson, A.: Water-scrum-fall is the reality of agile for most organizations today. Forrester Research **26**(2011), 1–17 (2011)
39. Xu, P., Ramesh, B.: Software process tailoring: An empirical investigation. Journal of Management Information Systems **24**(2), 293–328 (2007), `http://www.jstor.org/stable/40398686`
40. Zacarias, M., Martins, P.V., Gonçalves, A.: An agile business process and practice meta-model. Procedia Computer Science **121**, 170–177 (2017)