PRiM: an i*-based process reengineering method for information systems specification

Gemma Grau*, Xavier Franch, Neil A.M. Maiden

*Universitat Politècnica de Catalunya (UPC), c/ Jordi Girona 1-3, Barcelona E-08034, Spain.

Centre for HCI Design, City University. Northampton Square, London EC1V OHB, UK.

Abstract

Information system development can often be addressed as a business process reengineering practice, either because it automates some human-based processes or because it replaces an existing legacy system. Therefore, observing and analysing current processes can enable analysts to converge on the specification of the new system, generating and evaluating new system alternatives throughout. In this paper we propose a method to support this reengineering process that analyses the strengths and weaknesses of the current process; considers the strategic needs of the organization; provides guidelines for the prescriptive construction of i* models; and drives the systematic generation and evaluation of alternative technological and organizational solutions for the new system.

Keywords: Goal modelling; i* modelling method; Process reengineering; Requirements elicitation; Use case; Generation and evaluation of alternatives, System analysis; System design

1. Introduction

Developing information systems seldom takes place from scratch. New information system often automate tasks that previously undertaken by humans in an organization, or substitute systems that are becoming obsolete from the organizational point of view. Consequently, in most cases, we can consider information systems development and business process reengineering as two views of the same activity that need to be reconciled.

From the business process reengineering perspective, the specification of a new system starts from observation of the current process, understanding the rationale behind decisions, formulation of new processes or enhancements to existing ones, generation and evaluation of alternatives and, finally, construction of the detailed target specification itself. To deliver this process, the intermediate stages would benefit from modelling techniques with which to undertake a strategic analysis of the current process, the evaluation of its strengths and weaknesses, and the systematic generation of new system alternatives.

The use of the i* framework in business process reengineering [48] provides an appropriate context where the rationale behind the current process design is modelled by means of intentional concepts. The resulting i* model is the basis for searching and evaluating process alternatives and for obtaining the specification of the new system. Although there already exist several proposals that obtain detailed system specifications from i* models [5, 11, 28, 42], the problem of creating an i* model in a prescriptive way is not often addressed, and the reengineering process is at risk because of a failure to search and evaluate alternatives systematically.

In this paper we propose PRiM, a Process Reengineering i* Method, that addresses system development as a process reengineering exercise, focusing on how to build i* models prescriptively and how to generate and evaluate alternatives in a systematic manner. Well-established requirements

1 Corresponding author. Tel.: +34-93-413-68-72; fax: +34-93-413-78-33. E-mail address: gggrau@lsi.upc.edu.
engineering techniques and artefacts are used to obtain requirements in an effective way, describe implicit knowledge, and study and document processes. The ultimate goal is to use i* to model current processes in a prescriptive way and to drive the reliable use of patterns and evaluation techniques for the generation and evaluation of new system alternatives.

We argue that the application of the reported reengineering framework has the following benefits: (i) modelling the current process can provide more complete knowledge about how the organization works and stakeholders’ current expectations; (ii) analysis of the resulting model can supply the rationale behind the current design, and highlight the practices that have to be changed and why; and; (iii) the reasoning capabilities of i* guide the exploration and evaluation of different technological and organizational solutions in a systematic manner driven by the strategic needs of the organization. As a result, PRiM can be applied not only to process reengineering but also to design systems with legacy software and as a starting point for developing a product line.

The paper is organized as follows. Section 2 provides an introduction to the main concepts of the i* language. In section 3 we present and overview of the PRiM method, which is explained in detail in sections 4 to 9. Tool support and related work are explained in sections 10 and 11. In section 12 we analyse the different techniques used in PRiM, whilst in section 13 we discuss the experimentation of the method. Finally, in section 14, we present the conclusions.

2. The i* Language

The i* framework proposes the use of two types of models for modelling systems, each one corresponding to a different level of abstraction: the Strategic Dependency (SD) model represents the intentionality of the process and the Strategic Rationale (SR) model represents the rationale behind it.

A SD model consists of a set of nodes that represent actors and a set of dependencies that represent the relationships among them. Dependencies express that an actor (depender) depends on some other (dependee) in order to obtain some objective (dependum). Thus, the depender depends on the dependee to bring about a certain state in the world (goal dependency), to attain a goal in a particular way (task dependency), for the availability of a physical or informational entity (resource dependency) or to meet some non-functional requirement (softgoal dependency).

Actors can be specialized into agents, roles and positions. A position covers roles. The agents represent particular instances of people, machines or software within the organization and they occupy positions (and as a consequence, they play the roles covered by these positions). The actors and their specializations can be decomposed into other actors using the is-part-of relationship.

A SR model allows visualizing the intentionality behind an actor by means of a hierarchy of intentional elements delimited by its boundary. Thus, it provides the SD model with reasoning capabilities. The dependencies of the SD model are linked to intentional elements inside the actor boundary. The intentional elements of the SR model can also be goals, tasks, resources and softgoals; and are decomposed according to these types of links:

- Means-end links establish that one or more intentional elements are the means that contribute to the achievement of an end. The end is usually a goal, whereas the means is usually a task. There is a relation OR when there are many means linked to the end, which indicates that there are different ways to obtain that end.
- Contribution links are Means-end links with a softgoal as an end. They allow to state explicitly if the contribution of the means towards the end is negative or positive.
- Task-decomposition links represent the decomposition of a task into different intentional elements. There is a relation AND when a task is decomposed into more than one intentional element.

The graphical notation is shown in Fig. 1, as an example, a traditional learning system. On the left-hand side, we show the SR model of the actor Teacher, and within its boundary we remark the relationship among its intentional elements. On the right-hand side, we show the SD dependencies between the Teacher and a Student.

For more details about i*, we refer to [48]. A UML conceptual model of the constructs used in this paper may be found at [3].

![Fig. 1. Example of an i* model for a traditional learning system.](image)

3. Overview of the Method

The PRiM method is applied to business process reengineering as presented in [48], where an i*-based
framework consisting of four different phases is defined by: (i) modelling the process based on its intentional concepts; (ii) a systematic search for process alternatives; (iii) a systematic evaluation of process alternatives with respect to stakeholder interests; (iv) the possibility of connecting strategic reasoning with information system development.

PRiM uses this reengineering framework as the starting point but extends it in the following ways: a) we add a preliminary phase for domain information gathering in order to obtain the description of the current process; b) we focus on how to build an initial $i^*$ model from this information in a prescriptive manner; c) we provide some techniques for discovering new strategic needs; d) we drive the systematic generation of process alternatives; and, e) we propose a framework to drive the evaluation of the generated alternatives.

All of these contributions have a similar aim, namely to treat business process reengineering as a systematic process, and therefore reduce the inherent uncertainty in $i^*$-based reasoning. Our final objectives are to generate the specification of the new system based on the organization’s strategic goals, to explore alternatives and choose the most convenient organizational or technical solution, and to do this in a highly reliable and effective way. To achieve these objectives, the reengineering method applies rules, guidelines, patterns and questions that, together, articulate a well-defined process.

Fig. 2 presents an overview of PRiM. PRiM is composed of six phases. In the first phase, the current process is analysed using several requirements engineering techniques, and a descriptive model is built based on results from observation. In the second phase, the $i^*$ model is constructed to describe the behaviour and rationale of the current process. The reengineering analysis is done in the third phase using the $i^*$ model to discover new strategic needs. The systematic generation of alternative solutions is done in the fourth phase through the addition of new actors and the reallocation of responsibilities between them. Those different alternatives are evaluated in the fifth phase in order to select as the best-fit candidate solution. Finally, the specification of the new system can be automatically generated from the $i^*$ of the selected solution. In the rest of the paper, we describe each of these phases.

4. Phase 1: Analysing the Current Process

The first phase of the PRiM method captures and records information about the different elements of the current business processes in a social or socio-technical system. Several techniques can be applied to obtain this information, including techniques used in the RESCUE process [28, 29], a requirements process that combines techniques from human computer interaction and requirements engineering to gather data and describe the current process in its wider context of use.

4.1. The RESCUE Method

Human activity modelling represents the situated behaviour of human actors in the process, including the cognitive behaviour of an individual actor and the social behaviour of a group of actors. The purpose of human activity modelling is to understand the current process and its context of use, in order to baseline and inform changes specified in the specification of the new system. Data gathering techniques include observation of the current process and, if available, software system use reports, informal scenario walkthroughs, and interviews with stakeholders.

Human activity modelling produces two important deliverables that baseline the analysis of the current process and the changes to it. Context Models extend data-flow diagrams with notations for different levels of system actor.

---

**Fig. 2.** Overview of the PRiM methodology, showing the steps undertaken in each of the phases.
Their concentric circles around the new software system actors provide a simple but powerful approach for modelling system boundaries. Each model describes the system actors, their involvement in the current process and data flows between them. Human Activity Models (HAMs) are rich, structured descriptions of the current behaviour that actors undertake to achieve a goal. Each HAM is modelled according to a predefined structure that enables analysts to collect data from the current process in a systematic manner. Example fields in a human activity model include actors’ goals, triggering events, preconditions, assumptions, constraints, normal and alternative courses of actions and resources. Each HAM is represented using structured natural language, in order to facilitate a more expressive description that captures the richness of the different contexts, the behaviours and the resource used in the current process.

4.2. Documenting the current process

In Phase 2, captured information about the current business processes is used to build the i* model. As this information can be obtained by using several different techniques, the resulting artefacts may not share the same structure. These differences may cause difficulties in applying the rules and guidelines provided in the next phase and, to avoid it, we propose to use an intermediate template for documenting the process and unifying the structure of the information.

Scenarios are considered an effective technique for requirements engineering documentation and process analysis. Although there are several proposals for writing scenarios [41], we recommend to use our own simplified notation for process scenarios, so-called Detailed Interaction Script (DIS). At this point, all the scenario information needed has already been obtained, therefore each DIS is just an intermediate template with which to organize the information to facilitate the further i* model construction. The DIS template includes goals, actors, preconditions, triggering events and postconditions, which can be directly obtained from the process documentation, which in this case are HAMs. DIS actions are the HAMs atomic actions written in a more structured way. Thus, for each action we state: the actor who initiates the action, a short description of the action and, if the action produces or consumes a resource, the resource involved. If the action requires interacting with another actor, the actor addressee is also stated. Some examples of DIS are reported later in the paper.

4.3. Documenting a Collaborative Exercise Case Study

To illustrate our method we applied PRiM to a case study in which we designed a new software system for completing an on-line Collaborative Exercise. The main objective of the case study was to adapt traditional courses on-campus to the internet technologies and, therefore, permit students to perform part of the courses off-campus. Specifically, we focus assisting the students in completing an on-line Collaborative Exercise.

In this first phase, we analyse the current process, which we assume is purely social and involves a Teacher and a group of Students. In order to document it, we divide it in several activities that are described using human activity modelling techniques. Fig. 3 shows an UML activity diagram [45] with the identified activities for completing the Collaborative Exercise. Each activity consists of a set of related actions which are closely related in nature and in time. As all the activities are part of the same process, some precondition relationships are established between them.

![Activity diagram of the activities performed to complete a collaborative exercise in a traditional learning process.](image)

In order to document the processes performed in this activities we fill up the corresponding DIS template. In Table 1 we can see the DIS of the activity Organization of a Collaborative Exercise. We can observe that the resources provided by both the Teacher and the Student, such as Provide several alternative schedules, Analyse student schedule preferences or Decide definitive schedule are made explicit. In the DIS we differentiate among the produced resources (which are the ones produced during the action described in the row) from the provided resources (which are not directly produced by the action, but have to be provided in order to accomplish it). Once the Teacher has decided the Definitive Schedule based on the Student Preferences, it delivers it to them.

Once the time scheduled for the exercise is established, the Collaborative Exercise can be executed. Table 2 shows the DIS for this activity. The Teacher splits the Students...
into groups and provides the exercise problem statement; the Student executes the exercise in a collaborative way by interacting with the other Students. We can observe that, in order to perform those collaborative actions, the Student depends on another Student for producing a certain resource. In that specific case, the DIS action row shows a produced resource which is the one produced jointly with the other Student.

5. Phase 2: Building the $i^*$ Model of the Current Process

The reliable generation and evaluation of alternatives requires that the $i^*$ model of the current process is developed in a systematic and prescriptive way, in order to avoid taking decisions upon incomplete or ill-structured knowledge. Ideally, different people modelling the same process in the same organization should be able to obtain similar results. However, it is possible to obtain different models for the same process if it is analysed on different
organizations because, since \( i^* \) is a highly intentional modelling technique, the strategic reasons of every organization may change, and therefore the resulting models may be different. Thus, in spite of the modelling process that is followed, the organizational context plays an important role. As it is mentioned in [31], there is evidence that ignoring the difference between organizational realities and process logistics often causes a mismatch in the process analysis. Therefore, as proposed in [1], we distinguish between two different kinds of goals: descriptive goals that can be identified using analyses of current processes, and prescriptive goals that are provided by strategic management.

Using these two different types of goal we build the \( i^* \) SD model in two steps to distinguish the operationalization of the process (dealing with descriptive goals) from its strategic intentionality (prescriptive goals). The result is an \( i^* \) model with two different parts: the Operational \( i^* \) Model (mainly composed of resources, tasks and some goals) and the associated Intentional \( i^* \) Model (which adds goals and softgoals to the Operational one). To enable the models to be constructed and add rationale to our analysis of the current process, both SD and SR \( i^* \) models are developed. The resulting \( i^* \) model can then be checked for consistency, as proposed in the RESCUE process [28].

5.1. Step 2.1: Actor Identification and Modelling

The first \( i^* \) modelling step is the identification of the actors and their intentionality (i.e., its main goal). The actors are identified from documentation of the current process obtained in the previous phase. Each stakeholder is modelled as a different \( i^* \) actor and a single actor represents the software system if it already exists. If the system is considered to be too large or has well-differentiated parts, we may decompose it using the is-part-of construct provided by the \( i^* \) framework. Other aggregation and specialization relationships between the actors can be identified and included in the model. Due to the unavoidably iterative nature of the process, if new actors appear in later steps, a further iteration of Step 2.1 is needed in order to take them into account.

The actors identified for the Collaborative Exercise case study are a Teacher (Collaborative Exercise Evaluated) and a Student (Collaborative Exercise Completed).

5.2. Step 2.2: Building the Operational \( i^* \) Model

In order to be prescriptive when building the \( i^* \) model at the operational level, we use both the SD and the SR models. To obtain dependencies in the SD model we analyse each of the activities identified in Phase 1 (see section 4) focusing on the actions stated in the DIS of each activity.

The benefits of using the DIS for analysing the activities are two-fold. Firstly, the analysis of a piece of process in a chronological way can be easy to perform and tends to yield similar results even if performed by different people. Secondly, it is possible to translate the information of the table to the \( i^* \) model we are building, by applying the following rules:

- **Rule 1: Modelling activity-tasks.** Every activity in which an actor is involved is modelled as a task in its SR. This task (hereafter called an activity-task) is associated to the actor main goal (already identified in step 2.1) using a Means-End link. Activity-tasks are named after the activity they are related to. Once all the activity-tasks are modelled, we obtain a first level of decomposition on the SR model of each actor.

In Fig. 4 both the Teacher and the Student, which are the two actors involved in the activity Organize a Collaborative Exercise, have an activity-task with the name **Organize a Collaborative Exercise** in their SR decomposition.

- **Rule 2: Modelling action-tasks.** Every activity-task is decomposed into the actions of the DIS corresponding to its activity. This is done by translating each action into a task (hereafter, action-task) and relating this action-task with a task-decomposition link to the corresponding activity-task of the action initiator.

In Fig. 4 the activity-task Organize a Collaborative Exercise of the Teacher actor is decomposed into three action-tasks: **Provide Several Alternative Schedules**, **Analyze Student Schedule Preferences**, and **Provide Definitive Schedule**. On the other hand, as the Student actor is the action initiator of two of the actions of this activity, its activity-task is decomposed in two action-tasks: **Analyze Alternative Schedules** and **Provide Schedule Preferences**.

- **Rule 3: Modelling actions that produce/provide resources.** If the action on the DIS produces or provides a resource, this resource becomes a resource dependency where the action addressee is the dependee and the action initiator, the depender. In the SR model, this...

![Fig. 4. Piece of the \( i^* \) operational model, concerning the dependencies derived from the DIS Organization of a Collaborative Exercise.](image-url)
dependency is linked to action-task that produces or provides the resource.

In Fig. 4 the Student depends on the Teacher for the produced resource Alternative Schedules, and is the dependee for the consumed resource Schedule Preferences. The Student also depends on the Teacher for the produced resources Exercise Schedule however, at this point of the analysis, it is not possible to establish the action which consumes this resource, and therefore the dependency is related to the actor boundary.

- Rule 4: Modelling actions that consume resources. If the action on the DIS consumes a resource, this resource has to be produced by some other action. Thus, we look for those dependencies produced by other actions and assigned to the actor. If the resource has already been produced, we link it to the specific action-task. If not, we link directly to the actors’ activity-task.

In Fig. 4 we can observe that the action-task of the Student Analyse Alternative Schedules consumes the resource Alternative Schedule, whilst the action-task of the Teacher Analyse Student Preferences consumes the resource Schedule Preferences.

- Rule 4: Modelling Reflexive Actions. The actors used in the DIS and in i* correspond to roles rather than specific agents. This is why, sometimes, a role may depend on the same role for performing a certain action, it is what we call reflexive actions because the DIS action initiator and action addressee correspond to the same actor. To model these actions in i*, we first create a new actor with the name of the role followed by the word ‘Group’, and with the same main goal as the role. Each reflexive action is modelled as an action-task in the task decomposition of both the actor and the actor group. In order to represent the collaborative nature of the action, the produced resource is split into two different resources: the first one is preceded by the word ‘Individual’ and the second one by the word ‘Group’. The group actor depends on the individual resources, and produces the group resources that are consumed by another actor (it can be the individual or not).

For instance, in our case study, we create the actor Student Group with the goal Collaborative Exercise Completed. Therefore, in Fig 5 it is possible to see several examples of modelling reflexive actions. For instance the action Solve Exercise is modelled as two different action-tasks, one in the i* SR model of the Student actor and the other in the i* SR model of the Student Group actor. The resource Preliminary Results that is produced by the action is split into two different resources: Individual Preliminary Results (where the Student is the dependee and the Student Group the depender) and Group Preliminary Results, where the Student Group is the dependee and the action-task Discuss Preliminary Results of the Student is the dependee. However, the dependee of the resource produced by the group may not be the individual, that is, the group is the dependee and the individual is the depender.

- Rule 5: Modelling alternative course of actions. Actions in alternative courses are also modelled as actions-tasks but, as they are not always undertaken, they are linked

Fig. 5. Piece of the i* operational model, concerning the dependencies derived from the DIS Execution of a Collaborative Exercise.
with a means-end link to the action-task that causes the alternative course. If it is not related to a specific action-task, the means-end link is directly linked to the activity-task. Resources produced, provided and consumed are considered as in rules 2 and 3.

In Fig. 5 we can see that the action-task Ask Doubts has been added as means-end decomposition of the tasks Solve Exercise and Discuss Preliminary Results on the Student SR model. As the alternative action does not directly involve an action-task of the Teacher, it is linked to the activity-task Execute a Collaborative Exercise. Finally, the Student produces the resource Question, which is consumed by the Teacher, who consumes it to produce the resource Answer.

- **Rule 6: Modelling preconditions, postconditions and triggering events.** Every precondition, triggering event and postcondition of the activity has to be explicitly modelled. As they represent the achievement of a certain state, they are modelled as goals in the i* model. Preconditions and postconditions are added to the activity-task as task-decomposition goal elements. Triggering events are modelled as goal dependencies where the actor who initiates the activity-task is the depender and the one that undertakes the triggering task is the dependee.

In Fig. 5 the postcondition of the activity Execution of a Collaborative Exercise is modelled as the goal Collaborative Exercise Results Delivered on the SR model of the Student. The Teachers’ action-task Provide Definitive Schedule triggers the activity-task Execute a Collaborative Exercise, and thus, the Student has a goal dependency on Collaborative Task Scheduled. The activity-task Execute a Collaborative Exercise also consumes the resource Exercise Schedule.

All the proposed rules can be performed systematically and even automatically with appropriate tool support, and there is only one aspect in which the process is not prescriptive. Rules translate resources to resource dependencies but, sometimes, task dependencies are more suitable. To differentiate among them we it is necessary to consider what is more important to the dependee — the resource involved (resource dependency) or the way the resource is obtained (task dependency).

### 5.3. Step 2.3: Building the Intentional i* Model

The Intentional i* Model complements the Operational i* Model by representing the intentionality behind the analysed process. As it depends on the organization strategic needs its construction entails more freedom. This is the reason why the Intentional i* Model is only composed by goals and softgoals. The application of goal-oriented requirement engineering methods [40] helps to discover the intentionality in a systematic way. Thus, our proposal use the following set of guidelines, which are applied sequentially, intertwined or iterated whilst new goals or softgoals are still being discovered:

- **Guideline 1: Analysis of the Activities Intentionality.** An initial set of prescriptive goals is obtained directly from the current process. Assuming that each of the studied activities represents the achievement of a goal, strategic goals are obtained as a response to the following question:
  - Which is the final state that is achieved by executing the activity?
  - Which is the actor that needs to attain the goal?
  - From which actor does it depend to obtain the goal?

In the Collaborative Exercise case study, the execution of the activityExecution of the Collaborative Exercise leads to the achievement of the final intentional state Collaborative Exercise Passed where it is the Students who depend on the Teacher to achieve it. We remark that this goal is not the postcondition of the activity, which is the goal Collaborative Exercise Results Delivered. Therefore, we model the activity-task Execute a Collaborative Exercise as a means for obtaining the goal Collaborative Exercise Passed and we state a means-end relationship between these two intentional elements. In Fig. 6 we can observe that we have reorganized the hierarchy of the SR model in order to represent this insertion in the i* SR model of the Student. Consequently Collaborative Exercise Passed becomes a means to achieve the goal Collaborative Exercise Completed. Moreover the activity-task Organize the Collaborative Exercise is also a means to achieving the state Collaborative Exercise Passed, so a means-end is also specified between the two.

- **Guideline 2. Decomposition of the Intentional Goals and Softgoals.** This initial set of descriptive goals and the already existing Operational i* Model goals are the basis for obtaining further new goals. A directed inquiry analysis of the process reveals that goal-based [38] and stakeholder interview techniques can be applied [18]. Based on these proposals, we analyze the Operational i* Model with the stakeholders and formulate some questions over the i* SR intentional elements. These questions address intentional issues related to the achievement of the goal such as the why, what, how, who and where.

For instance the goal of the Student Collaborative Exercise Passed can be decomposed by answering the question:
  - What exactly has to be done to achieve that goal?

As an answer we obtain the goal Exercise Finished within Schedule Time (where the dependee is the Teacher) and the softgoal Provide Correct Results (where the depender is the Students Group). This softgoal can be further decomposed by answering the question:
  - What exactly has to be done to get the exercise correct results?
As the conditions Be in a Good Group and Achieve Agreement on the Results contribute to the softgoal, the softgoal is decomposed into two new softgoals corresponding to these conditions. The dependee for Be in a Good Group is the Teacher, whilst the one for Achieve Agreement on the Results is the Student Group (see Fig. 6).

- **Guideline 3: Quality-Attributed Analysis of the Dependencies Intentionality.** The classification of goals according to some attributes helps us to manage these goals. For instance, in KAOS [10], goals are classified into satisfaction, information, robustness, consistency, safety and privacy goals; and likewise classifications are applied in the NFR framework [8]. Based on this underlying principle, we propose to use a quality attributes catalogue, and specifically the ISO/IEC 9126-1 [26], to generate questions automatically. Quality attributes may be related to the type of intentional element: for instance, if the analysed element is a resource, we can ask about data security or data accuracy whilst if it is a task, questions about efficiency or usability are more appropriate. Most of the quality attributes refer to important goals and softgoals, but we are only interested in the most crucial ones. Thus the questions are formulated in terms of how critical is the attribute for the element of the process, and can be applied to both the dependums and the intentional elements of the SR model. The concrete writing of questions can be associated with the catalogue itself, therefore providing a systematic way to generate goals and softgoals.

For instance, in the Operational i* Model Collaborative exercise case study, the resource Exercise Statement can be analyzed by asking the questions:

- If the Exercise Statement is not accurate enough, will the process fail? Which actors will be affected?
- If the Exercise Statement data privacy is violated, can someone get hurt or damaged? Which actors will be affected?

In Fig. 6 we state that the accuracy on the Exercise Statement is crucial to execute the exercise, and therefore we introduce a softgoal Exercise Statement be Accurate. The Student depends on the Teacher for ensuring it. However data privacy is not so critical in general, neither the teacher nor the Student have any dependency about that (we consider that the teacher uses each exercise statement only one time, and once it is given to the students, it can be disclosed).

Questions can also be applied in a similar manner to the SR model. In Fig. 6, the action-task Discuss Preliminary Results is further decomposed into the softgoal Agreement Found as Soon as Possible, as an answer to the question:

- If Discuss Preliminary Results is not performed efficiently in time, can the process fail?

- **Guideline 4: Analysis of Contributions and Conflicts.** The resulting set of goals and softgoals can be analyzed in order to identify contributions and conflicts between the different intentional elements by means of the i* contribution links. This analysis can be done as proposed in the NFR framework [8] or by considering the relationships between quality attributes already stated in the quality model [12].

In Fig. 6, the softgoal Agreement Found as Soon as Possible contributes positively to the softgoal Exercise Finished within Schedule Time.

![Fig. 6. Piece of operational and intentional SR i* model for the activity Completion of a Collaborative Exercise.](image)
5.4. Step 2.4: Checking the i* Model

The intentional elements and dependums of the Intentional i* Model can be checked for the consistency of the application of the applied rules and guidelines. Fig. 7 presents a meta-model with the simplified concepts of the three models used: HAM, DIS and i* models. The baseline concept mappings across those models are defined with thicker horizontal lines. Thus, meta-model maps actor goals in HAM to conditions in DIS and i* goals and soft goals. As the DIS is a structured version of HAM, there is a direct mapping between the concepts of activities, actions and actors between both models, whilst the resources in DIS can be of three types: produced (p), consumed (c) or provided (pv). DIS activities are modelled into i* tasks and actions are mapped into resources or tasks in the Operational i* Model.

Check 1.1. Every activity on the HAM should correspond to one or more activities in the DIS.
Check 1.2. Every human activity goal should correspond to one or more conditions (goals or postconditions) in the DIS.
Check 1.3. Every precondition and assumption on the HAM, should correspond to preconditions in the DIS. Triggering events are modelled equally in both diagrams.
Check 1.4. Every actor of the HAM is mapped into one or more actors in the DIS.
Check 1.5. All actions on the HAM normal course and alternatives courses should correspond to actions in DIS detailing the action initiator and, if required, also the action addressee.
Check 1.6. All resources appearing in the HAM actions, should be detailed as produced, provided or consumed resources in the DIS.

At the second stage, cross-checking is done in order to ensure the correspondence between the DIS and the resulting i* models. Those checks are:

Check 2.1. Every DIS activity is modelled as a task, called activity-task.
Check 2.2. Every main goal of an actor is means-end decomposed into those activity-tasks where the actor performs an action.

Check 2.3. Every normal course action inside an activity is modelled as a task, which decomposes the corresponding activity-task on the SR model of the actor that initiates the action.
Check 2.4. Every provided resource involved in an interactive action, appears as an SD resource dependency or task dependency between the actors involved in the interaction, where the dependee produces the resource and the dependor consumes it.
Check 2.5. Conditions of the activity are modelled as SR goals (for preconditions and postconditions) and goal dependencies (for triggering events).
Check 2.6. Each activity-task is means-end decomposed into its main intentional goal, which can be refined into other goals.
Check 2.7. Some non-functional constraints are stated over the resources and the task, leading to softgoals both in the SR and the SD model.

6. Phase 3: Reengineering the Current Process

As the purpose of the method is to reengineer a business process, a strategic analysis is needed to introduce new issues and improvements to that process. Thus, stakeholders may want to improve some aspects of the current process, and this improvement might be manifest as the introduction, change or deletion of goals and softgoals. KAOS [10] proposes to drive this process by applying patterns that have an impact on the system behaviour: Achieve and Cease goals generate behaviours, Maintain and Avoid goals restrict behaviours, whilst Optimize goals compare behaviours. We use these patterns for analysing the SD i* model with the stakeholders as following:

1. Firstly we restrict the behaviour of the current process by analysing the intentional goals and softgoals. We classify them into two groups: the goals and softgoals that we want to Maintain, and the goals and softgoals that we want to Avoid. If a goal has to be avoided, new goals and softgoals arise in order to state how their behaviour has to be avoided.

2. This new set of goals is analysed in order to generate new behaviour. Thus, we search for new goals that we want to Achieve or old ones we want to Cease. If the new goals to Achieve involve the addition of new activities to the process, a DIS is created for the analysis of the process and the i* model is completed with new SD dependencies and SR intentional elements by applying the steps on Phase 2. A goal can only be Ceased if it does not affect the achievement of another goal. In terms of activities, this means that if some of the actions it involves are preconditions or triggering events of another activity, it cannot be removed unless the other activities are also removed.

3. Finally, Optimize goals are added in order to compare the behaviour. Questions such as the ones proposed in step 2.3 in section 5.3, have to be applied.

For example, in the Collaborative Exercise case study, all the goals are maintained except the goal Share location during the exercise (see Fig. 5) that is considered to be
To find the software roles best suited to the new solutions and configurations of爆炸, software roles are also likely to arise. As a consequence, Phase 3 has to be applied again, and the generation of alternatives is restarted. If new reengineering issues arise during the generation of alternatives, they have to be added to the Reengineered i* Model.

7. Phase 4: Generation of Alternatives

One of the main strengths of goal-oriented modelling is its adequacy for exploring different ways to achieve strategic aims. This can be seen in many proposals. The i* framework itself [48] seeks systematic searches for process alternatives by using means-end reasoning and hierarchical decomposition of tasks into their intentional elements. The TROPOS project [5] defines the architectural organization of the system by exploring alternatives whilst introducing new actors. Those actors are defined according to the choice of a specific architectural style and the benefits that they provide for the fulfillment of some specific functional and non-functional requirement. In the KAOS approach [10] the identification of alternative responsibilities and the assignment of actions to responsible agents is an important feature of their goal-directed acquisition strategy.

Taking these approaches as the starting point, we use the Reengineered i* Model obtained in the previous phase as a basis for obtaining new alternatives that are also modelled in i*. Alternatives are generated by adding new actors to the system and reallocating responsibilities between them. If new reengineering issues arise during the generation of alternatives, they have to be added to the Reengineered i* Model. As a consequence, Phase 3 has to be applied again, and the generation of alternatives is restarted.

7.1. Step 4.1: Adding New Actors to the System

The addition of actors to the system is done by means of exploring new roles to fulfill in the business process. It is useful to analyse current solutions to similar processes such as organization structures or software solutions, so that new actors arise from rational analysis rather than combinatorial explosion.

One possibility is to apply organizational patterns to explore the application of well-known solutions that define social patterns and organizational styles expressed as configurations of i* concepts [32]. As the intention is to design a new process, software roles are also likely to arise. To find the software roles best suited to the new solutions we recommend using components catalogues or, even better, taxonomies of COTS components [2].

For each added actor, we need to decide its main goal. New actors are added in both Operational and Intentional Models. In Fig. 8 we have added to the i* model for the Collaborative Exercise case study an eLearning Software System actor. Its main goal is On-Line Collaborative Exercise Supported and, thus, the main goal of the other actors also has to change accordingly. Therefore, the Teacher’s main goal becomes On-line Collaborative Exercise Evaluated and the Student’s one, On-line Collaborative Exercise Completed.

7.2. Step 4.2: Reallocating Responsibilities

Once the new actors have been discovered, the existing dependencies must be reallocated. Each activity-task is analysed independently by asking the questions: Do we want this actor to keep satisfying the dependencies of the activity-task? Is there any other actor that can take that responsibility? Human capabilities may be checked and software component functionalities may be matched [15] to answer these questions, and depending on the answer, one of the following patterns is applied:

- **Pattern 1: Goal Achievement**. We consider that the responsibility for achieving the goal still falls on the current actor. As a result all of the dependencies related to the activity-task that operationalize this goal remain unchanged. In the Collaborative Exercise case study, one alternative is to let the Teacher decide the Exercise Schedule in the classroom so the responsibility for Organize a Collaborative Exercise still falls on the Teacher and the Student, which means that none of the dependencies related to that activity-task change (see Fig. 8).

- **Pattern 2: Goal Delegation**. We delegate the responsibility of achieving the goal. Thus, if actor A has an activity-task that operationalizes the goal and we decide that actor B will achieve it, we delegate the responsibility of the corresponding activity-task to actor B taking into account the following aspects:
  - A new activity-task with the same name is added to the Means-End decomposition of the main goal of actor B and all the dependencies related to their action-tasks are moved into it.
  - A new goal dependency is added stating that Actor A depends on Actor B for achieving the goal.

For instance, in Fig. 8 the responsibility of the task Execute a Collaborative Exercise goes to the eLearning System actor who handles the dependencies going to/stemming from the part of the SR model for the Teacher (see Fig. 5 for a representation of the previous model).

- The action-tasks of the reallocated activity-task are checked in order to ensure that actor B has all the knowledge and capabilities to undertake them. If actor B...
cannot fulfil them, we add a dependency to the actor
doing the task. In Fig. 8, we observe that the responsibility of Split
Students into Groups and Provide Exercise Statement
is now going to the eLearning System. However, the Teacher
has to provide the Group Assignment and the Exercise
Statement. Thus, we have two resource dependencies from
the eLearning System to the Teacher in order to obtain that
information, and two more dependencies from the
eLearning System to the Student in order to provide it. As it
can be observed in the case study, the eLearning System
mediates the interaction. To improve comprehension we
have renamed the action-task in this actor. Thus, Split
Students into Groups has been renamed to Provide Group
Assignment.

On the other hand we remark that, as the Student is
interacting with the eLearning system, all the Student
Group activity-tasks fall into the eLearning system because
the software is now the actor that mediates all the
collaborative activities. However, the Student Group actor
does not disappear from the model because the Student still
is having some intentional dependencies upon it.

- Finally, a goal dependency arises to model that actor A
depends on actor B to achieve its intentional goals on
the activity.

Thus, in the example, the Student depends on the
eLearning System to accomplish the goal postconditions of
the activity Collaborative Exercise Results Delivered (see
Fig. 8).

As we are in a reengineering context it can also be
possible that, for a certain alternative, some of the activities
require a different course of action (e.g. consideration of a
COTS component may require a predefined interaction
different from the existing one). On the other hand we have
to introduce new activities in order to operationalize the
goals and softgoals of the Reengineered i* Model. In those
cases, the following patterns can be applied:

- Pattern 3. Goal Operationalization. We add or change
the course of actions that operationalize a goal.

Sometimes the interaction between actors may be
different to the original one (e.g. the added actor can
perform tasks in a different order than the original
model, or the use of a software system actor constraints
the interaction with the other actors). In this situation,
the action-task in the existent activity-task needs to be
redefined in order to adapt to the behaviour of the new
actor. Furthermore, during the Reengineering Phase, it is
possible to add new goals to accomplish or avoid. As
these goals are not operationalized in the current
process, the DIS is not yet defined and its representation

![Fig. 8. Piece of i* model showing responsibility reallocation of the activity Performance of a collaborative exercise into the eLearning System actor.](image-url)
on the i* model has to be modelled. In both cases we recommend specification of the new course of actions in an auxiliary DIS template, then transfer the changes to the i* model following the rules provided in Phase 2.

- Pattern 4. Softgoal operationalization. Softgoals of the model can also be satisfied by operationalizing them. Some softgoals related to non-functional requirements, can also be operationalized into tasks, such as in the case of the softgoal ‘security accessing data’, which can be operationalized into a specific protocol such as a course of actions that asks a password to the user.

7.3. Step 4.3: Checking consistency between alternatives

Consistency checks are also applied in order to ensure certain equivalence between the Reengineered i* Model and the i* model of the generated alternatives. We remark that as the provided patterns may operationalize goals and softgoals in a different manner to the current process, new activity-tasks, action-tasks and dependencies arise in the i*.

In order to ensure that the i* models of the generated alternatives can be compared whilst having different structures we have to check that the intentional goal of the actors can still be accomplished either by the original actor or by some other actors. This can be ensured by checking the following properties:

- Intentional equivalence: Given two i* models M1 and M2, M1 and M2 are intentionally equivalent if:
  - M1 and M2 have the same intentional goals, being intentional goals the ones obtained in Step 2.3 (see section 5.3) and in Phase 3 (see section 6).
  - for each intentional goal G where an actor is dependee on M1, if G is operationalized in M1 it is also operationalized in M2; and if G is not operationalized in M1, it is neither operationalized in M2; and
  - for each intentional goal G where an actor is dependee on M2, if G is operationalized in M2 it is also operationalized in M1; and if G is not operationalized in M2, it is neither operationalized in M1.

- Intentional Inclusion: Given two i* models M1 and M2, M1 is intentionally included in M2 if:
  - M2 has, at least, the same intentional goals than M1, being intentional goals the goals obtained in Step 2.3 (see section 5.3) and in Phase 3 (see section 6).
  - for each intentional goal G where an actor is dependee on M1, if G is operationalized in M1 it is also operationalized in M2; and if G is not operationalized in M1, it can be or not operationalized in M2.

8. Phase 5: Evaluating Alternatives

The systematic evaluation of process alternatives has been addressed in the i* framework [48]. In the original proposal the SD model supports the systematic identification of stakeholders and their interests and concerns, whilst the SR model supports the systematic evaluation of alternatives through the concepts of ability, workability, viability, and believability. The Tropos project [5] also uses the i* capabilities in a similar way in order to connect strategic reasoning with information system development. As i* is a goal-oriented technique, some goal-oriented analysis methods, such as the NFR framework [8] and the AGORA method [30], can also be applied.

Therefore our proposal for evaluating alternatives is based on a structural analysis of the i* models as presented in the REACT method [13, 43]. Structural analysis is adequate in the context of PRM because of the characteristics of the resulting model. On the one hand, the prescriptive construction of the model ensures that the resulting i* model contains all the elements involved in the process and none of them is duplicated or omitted. On the other hand, alternatives solutions are constructed as alternative models whilst, in other approaches, the alternatives are modelled in the same model by using means-end links.

The structural analysis proposed in REACT is performed by evaluating properties considered of interest for the modelled system (such as security, accuracy or efficiency). Properties can be evaluated with structural metrics, which take into account the structural elements of the models (e.g., actors, SR elements, SD dependencies) and their relationships. Thus, structural metrics are defined in terms of the actors (actor-based metrics) or the dependencies (dependency-based metrics) of the model. The results of this analysis are used to inform the selection of the most suitable alternative. In order to guide the application of REACT we propose the following steps.

8.1. Step 5.1: Choosing Suitable Properties

Based on the structure of the i* SD models, it is possible to analyse their non-functional and organizational goals by means of several properties, and to use these properties to inform the evaluation and selection of alternatives. These properties are non-functional requirements or other process constraints that have already arisen in the previous phases.
of the method and, as such, they are modelled as non-functional goals in the i* models. However, not all the properties are equally important and, for the evaluation, we have to choose the most relevant ones by prioritising them (e.g., by considering individual stakeholder ranking of properties). As in Phase 2 we have obtained the intentionality behind the process by asking quality-directed questions, an analysis of which quality factors were more relevant on the current system also helps to establish the ranking of properties.

In the Collaborative Exercise case study, we can observe that many of the softgoals are associated with the properties: Ease of Communication (related to the softgoals Achieve Agreement on the Results, Be in a Good Group, and Deliver Correct Results), Process Agility (related to the softgoals Agreement Found as Soon as Possible, Exercise Finished within Scheduled Time), and Accuracy (Exercise Statement be Accurate, Final Results be Accurate). Thus, these properties are good candidates to be evaluated over the models.

8.2. Step 5.2: Defining Metrics over the Models

The properties can be evaluated with metrics defined in terms of the actors (actor-based metrics) and the dependencies (dependency-based metrics) of the model:

- **Actor-based metrics:** Given an architectural property \( P \) and an i* SD model that represents a system model \( M = (A, D) \), where \( A \) are the actors and \( D \) the dependencies among them, an actor-based architectural metric for \( P \) over \( M \) is of the form:

\[
P(M) = \frac{\sum_{a \in A} \text{filter}(a) \times \text{correctionFactor}(a)}{\| A \|}
\]

being \( \text{filter}(a) : A \rightarrow [0,1] \) a function that assigns a weight to the every actor (e.g., if the actor is human, software or from a specific kind), and \( \text{correctionFactor}(a) : A \rightarrow [0,1] \) a function that corrects the weight of an actor considering the dependencies stemming from or going to it.

- **Dependency-based metrics:** Given an architectural property \( P \) and an i* SD model that represents a system model \( M = (A, D) \), where \( A \) are the actors and \( D \) the dependencies among them, a dependency-based architectural metric for \( P \) over \( M \) is of the form:

\[
P(M) = \sum_{d \in D} \text{filter}(d) \times \text{correctionFactor}(a) \times \text{correctionFactor}(b)
\]

being \( \text{filter}(d) : D \rightarrow [0,1] \) a function that assigns a weight to the every depedem (e.g., if the depedem is goal, resource, task, softgoal if it is from a specific kind), and \( \text{correctionFactor}(d) : D \rightarrow [0,1] \) two functions that correct the weight accordingly to the kind of actor that the depender and the depeendee are, respectively.

8.3. Step 5.3: Evaluating the Alternative i* Models

Metrics are defined at the beginning of the evaluation phase and are used to evaluate all the alternatives generated in the previous phase. For instance, in the Collaborative Exercise case study, we define two properties: Ease of Communication and Process Agility, which are both evaluated by dependency-based metrics.

As stated in the formula above, for defining a dependency-based metrics, three factors have to be defined. Firstly, in order to weight the \( \text{filter}(d) \) we have to analyse the involved dependums according to the relevance they have for the corresponding property. For Ease of Communication, tasks are the dependums that better contribute to the property because they are more precise on establishing the way the element is provided. On the other hand, goals provide more freedom for achieving the dependum, whilst resources are more restrictive. Taking those criteria into account, we define:

\[
\text{filter}(d) = \begin{cases} 
1 & \text{if } d \in \text{Task} \\
0,8 & \text{if } d \in \text{Resource} \\
0,6 & \text{if } d \in \text{Goal} \\
0 & \text{otherwise} 
\end{cases}
\]

Correction factors are defined by assuming that Ease of Communication is better achieved when the actors are human, rather than software. So, we define:

\[
\text{correctionFactor}(a) = \begin{cases} 
1 & \text{if } a \in \text{Human} \\
0,5 & \text{if } a \in \text{Software} \\
0 & \text{otherwise} 
\end{cases}
\]

The metric for Process Agility is defined in a similar way. In that case we consider the dependums in terms of the agility they provide to the process. Thus, tasks are more constrictive than resources; and goals are the ones that provide more agility. As duplicated dependums within the model weaken Process Agility, the weight is divided by the number of times that the dependency appears in the model.

\[
\text{filter}(d) = \begin{cases} 
0,6/\#\text{occurrences}(d) & \text{if } d \in \text{Task} \\
0,8/\#\text{occurrences}(d) & \text{if } d \in \text{Resource} \\
1/\#\text{occurrences}(d) & \text{if } d \in \text{Goal} \\
0 & \text{otherwise} 
\end{cases}
\]

In terms of Process Agility, we can consider that software systems are more agile than humans, and that the Teacher actor is more trained in collaborative exercises and provides more agility than the Student actor. Finally, in human actors, the agility has less influence in the dependers that in the depeendees. Thus, we define:

\[
\text{correctionFactor}(a) = \begin{cases} 
0,9 & \text{if } a = \text{Teacher} \\
0,8 & \text{if } a = \text{Student} \\
1 & \text{if } a \in \text{Software} \\
0 & \text{otherwise} 
\end{cases}
\]
We remark that, in order to weight the filters and correction factors of the metrics, expert advice is strongly recommended. Metrics can also be defined in more or less detail depending on the desired accuracy and reuse of the results. This is a trade-off condition, because the more elaborate the metrics are, the more accurate but less easy they are define and reuse.

Once the metrics for the properties are defined, metrics can be evaluated. The evaluation consists of applying the different metrics on every alternative and to store the results for further analysis. In Fig. 9 we show three alternative solutions for the Collaborative Exercise case study. The solutions are modelled as structural representations, showing the kind and quantity of dependencies that appear among the actors. Softgoals are not shown in this representation because they are not considered for the analysed metrics. The candidate solutions are defined as follows:

A. As proposed in Step 4.2. (see section 7.2) the reallocation of responsibilities is done so that the activities Organization of a Collaborative Exercise and Evaluation of a Collaborative Exercise are undertaken without software support and Execution of a Collaborative Exercise is reallocated into the eLearning System, which acts as a mediator between the Teacher and the Student.

B. In the second alternative, all the activities are reallocated into the eLearning System, which acts as a mediator between the Teacher and the Student.

C. The third alternative is a variation of the second, in which the eLearning System is not only a mediator between the other actors, but it is also in charge of undertaking the following responsibilities that were undertaken by the Teacher in the second alternative: Decide the Exercise Schedule, Decide Group Assignment, Solve Student Doubts (e.g., by means of a FAQ list), and Correct the Exercise (e.g., by using a template introduced by the Teacher).

The evaluation of the properties Ease of Communication and Process Agility over these models provides the following results:

<table>
<thead>
<tr>
<th>Property</th>
<th>Alternative A</th>
<th>Alternative B</th>
<th>Alternative C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ease of Communication</td>
<td>0.519</td>
<td>0.427</td>
<td>0.401</td>
</tr>
<tr>
<td>Process Agility</td>
<td>0.419</td>
<td>0.378</td>
<td>0.446</td>
</tr>
</tbody>
</table>

We observe that the Ease of Communication decreases when the system undertakes more responsibilities, thus alternative A that involves direct communication between human actors is better than alternative C where the system not only mediates communication between humans, but also executes some of their responsibilities. On the other hand, Process Agility is enhanced in alternative C because the system has more responsibilities. However, when the system only acts as a mediator between the human actors such as in alternative B, the Process Agility scores lower than in the case where there is a direct contact between them (for instance, in alternative A) because in that case, direct communication is more agile.

We remark that if we reapply the evaluation with small variations on the values given when weighting the elements of the metrics, the evaluation results change but the ranking of suitability of the alternatives is maintained.

8.4. Step 5.4: Trade-off Analysis of the Results

The results of the evaluation must be analysed to decide the best candidate solution alternative. Reaching this decision can be challenging because it is not common that one alternative scores higher for all the evaluated properties. Thus, a trade-off analysis has to be done in order to achieve a reconciled solution. For instance in the Collaborative Exercise case study, none of the alternatives scores highest in both Ease of Communication and Process Agility properties. As a result trade-off decisions have to be taken by prioritizing one property over another (e.g., Process Agility rather than Ease of Communication, tends to favour the first alternative, whilst reversing the priorities tends to favour the second one). Furthermore the analysts might choose to adopt an intermediate solution (e.g. the first alternative provides the best Ease of Communication and still a good solution according to Process Agility).

![Fig. 9. Structural representations of the three i* models for the generated alternatives of the Collaborative Exercise case study.](image-url)
Although we have presented the steps of this phase in a sequential way, iteration is possible and new properties can be chosen and more accurate metrics can be defined in order to achieve a more precise evaluation. As a general rule, we recommend to begin the evaluation with coarse-grained metrics and to increase the level of detail when the results of the evaluation require a deeper analysis. The results obtained may also point out to new issues on the process redesign and, then, Phases 3 to 5 can be iterated.


The link between strategic reasoning and information system development has been widely addressed. Several current proposals provide guidelines for mapping an \(i^*\) model to an UML use cases and classes specification, including [5, 28, 42]. One more specific approach is the one proposed in [9], where guidelines are applied to obtain specific use cases scenarios for the Prometheus methodology.

The method proposed in [42] provides guidelines for constructing the use case specification from an \(i^*\) model that has already been built correctly. These guidelines obtain the actors and the use cases from the SD model, then use the SR model to obtain different courses of action. Table 3 shows the resulting use case obtained applying these guidelines [42] to our Collaborative Exercise case study.

10. Tool Support

The PR/M method involves the execution of different phases and the application of different methods and techniques. Although it can be applied manually, the use of tool support improves its applicability and, because of that, we have applied tools since the inception of the method.

Our first proposal is REDPEND-REACT [20], a graphical tool that assists the construction of complex \(i^*\) models by using a Visio stencil that allows to drag and drop the different \(i^*\) shapes (actors, goals, softgoals, resources, tasks and dependencies) into a Visio drawing page. Once the models are constructed, REDPEND-REACT supports the generation and evaluation of alternatives by using with the metrics proposed within the REACT framework [13, 43]. Therefore, the aspects of the PR/M method that are supported by REDPEND-REACT focus on the construction of the model and the generation and evaluation of alternatives. Despite being adequate for our earlier case studies, REDPEND-REACT is a graphical tool and, thus, as \(i^*\) models grow, it gets slower, models are difficult to manage, individual elements hard to locate, and the model layout has to be rearranged every time a new element is inserted.

In order to solve these drawbacks and provide more complete support for the PR/M method, we propose J-PR/M [23], a Java tool that we have developed within the Eclipse framework that runs over a MySQL database and is compliant with the \(i^*\) meta-model presented in [3].

<table>
<thead>
<tr>
<th>Use Case ID</th>
<th>UC2: Performance of an on-line collaborative exercise</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source</td>
<td>(i^*) model, activity-task: Perform a collaborative exercise</td>
</tr>
<tr>
<td>Actors</td>
<td>Teacher, Student, eLearning System</td>
</tr>
<tr>
<td>Problem statement (now)</td>
<td>The Teacher has organized an on-line collaborative exercise in order to evaluate her students. The Teacher splits the students into groups and provides them with the problem statement. The Student gets that information from the system and uses the system to discuss how to solve the exercise with the other students in her group. Once they obtain a final version they deliver it using the system.</td>
</tr>
<tr>
<td>Triggering event</td>
<td>The time for the on-line collaborative exercise has come.</td>
</tr>
<tr>
<td>Preconditions</td>
<td>The Teacher and the Student have previously logged into the eLearning system.</td>
</tr>
<tr>
<td>Assumptions</td>
<td>-</td>
</tr>
</tbody>
</table>

**Normal Course of Actions**

1. The Teacher splits the students into groups and introduces the group assignment to the System
2. The System provides the group assignment to the students
3. The Teacher introduces the problem statement to the System
4. The System provides the problem statement to the Students
5. The Student solves her exercise and introduces her preliminary individual results to the System
6. The System distributes all the individual preliminary results to the students of the same group
7. The Student provides individual comments for discussing the preliminary results
8. The System broadcasts the individual arguments to all the students of the same group
9. The Student provides agreement to the final results
10. The System delivers the group results to the Teacher

**Alternative Course of actions**

5a. The Student asks some doubts to the Teacher by means of introducing a question to the System
5b. The System made the questions available to the Teacher
5c. The Teacher gets the questions and provides answers to the System
5d. The System sends back the answers to the Student
The main difference between J-PRiM and other modelling tools is in the way the $i^*$ elements are introduced and visualized. Usually $i^*$ modelling tools represent the models as drawings (see [27], tool questionnaire section). However, J-PRiM does not show the $i^*$ elements in a graphical way, but in a tree-form hierarchy (see Fig. 10, left).

J-PRiM offers a different level of support for each of the steps of PRiM, according to the necessary amount of user involvement in each step. Therefore, the steps that require the user to enter data are supported by forms; steps that complement existent data but still require user expertise are guided; and steps that can generate new data without user interaction are automatically supported.

To support the first phase of the method J-PRiM allows a user to introduce the actors, resources and the activities of the process under study through forms. To support reuse, actors are stored in a catalogue. Resources and activities are specific of each project and activities are defined and stored following the DIS template (see Fig. 10, centre).

In the second phase, the construction of the $i^*$ model begins with the identification of the main goals of each actor. As PRiM provides prescriptive rules for generating the Operational $i^*$ Model from the DIS templates that are very specific, the tool generates this model automatically, applying all the checks provided in the method. Once the Operational $i^*$ Model is generated, the tool supports the application of the intentional guidelines in order assist the user on the definition of the Intentional $i^*$ Model.

Next, the current process is reengineered. J-PRiM allows the user to browse the goals and softgoals of the Intentional $i^*$ Model and add new goals and softgoals according to the patterns provided in [10] and adapted in the third phase of the method. The tool allows the user to state a rationale for each added element including who suggested it, and why.

In the fourth phase process alternatives are systematically generated by adding new actors to the system, removing existing actors, and reallocating the responsibilities between the others. The tool provides: 1) automatic generation of all the possible combinations according to some predefined filters; 2) assistance in order to help the user to apply the patterns that distribute the goal responsibilities between the actors. As the patterns still leave a certain degree of freedom when choosing the actors and reallocating the goal responsibilities, and to avoid

---

![Fig. 10. Screenshot of J-PRiM.](image-url)
combinatorial explosion and help the generation of the alternatives, the tool allows the user to document all actions in this phase fully using a traceability template. The template is based on the one proposed in [4] and states, for each alternative: the identifier of the alternative, an outline of the problem to be solved, an description of the strategy follow to solve the problem (pattern applied or set of decisions taken), an enumeration of the actors that are considered, and the transformations that have been applied to obtain the model by mentioned for each activity-task that the pattern has been applied to.

The generated process alternatives are evaluated in the fifth phase, where J-PR/M assists the user in the selection of properties and the definition of their metrics. As the defined metrics often rely on expert judgment and can differ according to the context and purpose for which they are applied, the tool allows the user to state the context, purpose and people involved in the evaluation. Properties and their metrics are stored in a Properties Catalogue in order to allow reuse, which allows the user to reuse existing properties and associated metrics or to create new properties. Once appropriate properties and metrics are selected, the tool automatically evaluates the models and presents a trade-off analysis to support the user in the selection of the most suitable solution. Finally, in the sixth phase, PR/M proposes the generation of the new system specification from the i* model of the chosen alternative. However, this feature is not yet supported by the tool.

To guide the application of the methodology, all the phases and their information are shown in tabs at the top of the tool working area (see Fig. 10) whilst their steps are shown in tabs at the bottom. To improve the usability of the tool, the interface avoids pop-up windows and shows the information of the forms mainly in the left section of the working area.

PR/M is considered as an iterative process that can be changed and refined by going back on each of its steps. Because of that, the tool allows the user to review the data introduced in a previous step at any time, then to reapply the method with new data. However, traceability over these changes is not stored by the tool unless the user manually saves them as different versions.

Finally J-PR/M also supports i* SD and SR modelling by introducing the elements individually. This can be done either when applying the PR/M method or when developing models from scratch. Changes over the i* model are executed by clicking on the element and choosing the corresponding action on the menu. For a comparison between J-PR/M and other i* modelling tools, we refer to [27], in the tool questionnaire section.

11. Related Work

The i* framework is used in a wide variety of contexts, however, much attention has been given to the analysis of its visual utility and the reasoning capabilities that it provides, rather than than the construction of the models themselves.

Since the i* framework is an agent-oriented language, the use of agent-oriented methodologies such as GAIA [46] may be considered to define i* models. However, most of the agent-oriented methodologies are not suitable for constructing i* models because, although they are build upon the same underlying concepts, they use their own models. So, it is not always possible to find equivalence between elements of these agent-oriented models and the elements provided by i*. An exception is the TROPOS methodology [5], which uses i* models on an agent-oriented behaviour.

Thus, as far as we know, the proposals that provide precise guidelines for constructing i* models are:
- The seminal proposal of the i* framework [48];
- The TROPOS methodology [5];
- The Goal-based business modelling oriented towards late requirements generation method [11];
- The RESCUE process [28, 29];
- A methodology for building i* models based on activities theory [37];
- A methodology for building i* models from BPEL Process Descriptions [44]; and
- The Ri/SD methodology [21].

TROPOS [5] is an agent-oriented methodology intended to support the entire software development process. TROPOS drives the construction of i* models in two steps. Firstly, goal-oriented requirements techniques are used to build the social system, then the software system is placed in the model to obtain the socio-technical system. As TROPOS relies on goal-oriented requirements techniques to obtain the models, the accuracy of the resulting models also relies on the accuracy provided by those techniques.

The Goal-based business modelling oriented towards late requirements generation method [11] uses goal-oriented techniques to obtain a so-called Goal-Refinement Tree. In the case study provided in [11], this tree presents a chronological structure and the correspondence between the goals and its goal refinement is comparable to the activities and actions identified in PR/M. However, this sequential order and its benefits are not explicitly mentioned and its construction relies on goal-decomposition strategies. The method defines steps for creating the social i* model from the Goal-Refinement Tree by first building the SD model then enriching it with the corresponding SR model. Once completed, some guidelines are proposed to build the socio-technical system business model. This is done by reassigning the SR tasks in a way similar to what we propose in PR/M (see section 7.2). Nevertheless, the method does not consider the generation or evaluation of several alternatives.

The RESCUE process [28, 29] is applied in the requirements specification stage of a project and employs five different techniques that are mutually supportive:
human activity modelling for the analysis of the current process; $i^*$ for modelling the system goals; systematic scenario walkthroughs for achieving use case modelling and the final specification; walkthroughs and scenario-driven impact analyses; and requirements management. The aim of the process is to elicit requirements through a bi-directional coupling of the $i^*$ model elements and the use cases and scenarios, allowing movement from one model to the other. Thus, once the $i^*$ model is constructed, the use case model can also be obtained.

The methodology presented in [37], uses activity theory as the starting point for building $i^*$ models. Thus, the activities and their actions are analysed to construct the model by applying concrete guidelines that map activity theory concepts to an SD model, and afterwards to an SR model. This approach is similar to the work reported in this paper, but differences [37] include building the SD model before the SR model, whilst PRM builds both models together. The activities and actions are also analysed differently because [37] does not provide any specific analysis of the resources involved in the actions, does not consider preconditions, postconditions or triggering events, nor any methodology for building the Intentional $i^*$ model.

In [44] several guidelines are provided in order to map BPEL Web Services descriptions into $i^*$ diagrams. The SD and SR models are developed simultaneously and the resulting models are automatically translated into the action language ConGolog in order to run simulations and analyse the model properties. This method relies on a very formal basis, but it can only be applied when web services BPEL descriptions are available.

Finally the R/SD methodology [21] builds the $i^*$ models

<table>
<thead>
<tr>
<th>Table 4. Summary and comparison of the $i^*$ methodologies according to the phases of PRM.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Methodology</strong></td>
</tr>
<tr>
<td>The seminal proposal of the $i^*$ framework [48]</td>
</tr>
<tr>
<td>The TROPOS methodology [5]</td>
</tr>
<tr>
<td>The Goal-based Business Modelling oriented towards late requirements generation method [11]</td>
</tr>
<tr>
<td>The RESCUE process [28, 29]</td>
</tr>
<tr>
<td>Building $i^*$ models based on activities theory [37]</td>
</tr>
<tr>
<td>Building $i^*$ models from BPEL Process Descriptions [44]</td>
</tr>
<tr>
<td>The R/SD methodology [21]</td>
</tr>
<tr>
<td>The PRM method</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
in two phases similar to TROPOS: one phase for the social system, the other one for socio-technical system. In R/SD [21] both phases may involve the partial or total construction of the SD and SR i* models and several guidelines are provided for decomposing the goals into sub-goals, and deciding the type of the dependums. Naming conventions for the intentional elements and procedures to control the size of the models are also stated. R/SD does not fit well in those cases where a software system is already present within the organization.

In Table 4 we summarize and compare these methodologies according to five of the phases of PR/M. Phase 3 is not included because only PR/M addresses the reengineering of the current system explicitly. Thus, those aspects that refer to each of the phases are described. We can observe that all the proposals use goal-oriented observation and specification techniques to obtain information about the current system. However, not all the proposals address the same issues when constructing the model: the proposals that focus on the operational level [11, 37, 44] provide precise guidelines for acquiring existing knowledge, whilst the ones that focus on the operational and the intentional levels [5, 21, 28, 29, 48] are less systematic in how the dependencies and intentional elements are obtained. Rather they use more reasoning-oriented approaches. Some of the approaches propose the complete construction of the SD model to guide the construction of the SR model [11, 37], however we argue that building both models together is more effective because the reasoning capabilities provided by the SR supports the modelling of the SD dependencies.

The generation of several alternatives is only addressed by [48] and PR/M, but some of the proposals build the i* model for the social system first to facilitate the construction of the socio-technical model by providing guidelines for inserting the system actor in the social system [5, 11, 21]. The evaluation of the generated solution is done by means of reasoning analysis [5, 48], or dynamically executing the model over some agent-oriented technology [5, 44]. Finally, most of the proposals offer the possibility of generating the specification from the i* model by providing their own methods or recommending other methods.

For another comparative analysis that analyses some of the mentioned techniques [5, 11, 37, 44, 21, 21] including PR/M, we refer to [22].

12. A Summary of PR/M Decisions

PR/M is a method that proposes to address information systems specification as a business process reengineering exercise predicated on the similarities between the activities in both fields.

As information systems specification involves requirements engineering and analysis techniques, the method has been constructed based on a rigorous state of the art analysis of both requirements engineering and business process reengineering techniques [24]. Based on this analysis we decided that i* is suitable to be used as a basis for a reengineering method and, so, in the design of PR/M we used as a starting point the 4 business process reengineering phases proposed in [48]. We added a first phase that makes the analysis of the current process explicit [19], to deliver a 5 phase method. In this first version, the reengineering of the current process and the generation of alternatives is done in the same phase, but after applying the method more times, two different phases were identified to adapt the number and objectives of the phases to the ones proposed in other reengineering methods [31].

There are several studies that prove that the representation of the problem affects its resolution [31]. We decided to use i* from amongst the modelling languages available for business process reengineering because:

- The i* framework is intended to address business process reengineering [48].
- The i* framework allows the representation of functional and non-functional requirements as well as business goals at the same level, thus bridging the gap that is usually found between requirements and organizational needs.
- The i* framework is goal-oriented, allowing the adaptation of other goal-oriented approaches to the method when needed.
- The i* framework is agent-oriented, providing a better view of the intentionality of the actors involved in the process.
- The i* framework has been successfully used for the generation and evaluation of alternatives.
- There are several approaches that transform i* into use cases and, as our main goal is to get the new system specification, this is an additional argument for its use.

In order to improve the correctness of the method, we designed PR/M by adopting existing techniques and artefacts instead of developing new ad-hoc techniques. The selection of these techniques was driven by their relevance and the authors’ expertise in applying them:

- **First Phase**: Human activity modelling is used in order to enrich PR/M with the experience of applying the RESCUE method [28]. The DIS template is designed taking into account the information obtained in other scenario-based methods [41].
- **Second Phase**: The i* model is built in two steps to distinguish the functionality performed by the stakeholders (descriptive goals) from their intentionality (prescriptive goals). These terms were proposed in [1] and adopted for other modelling techniques proposed in business process reengineering [31] and agent-oriented software engineering [46].
- **Third Phase**: KAOS [10] is one of the most wide-spread techniques for requirements elicitation [40]. As KAOS and i* are both goal-oriented, it is possible to adapt the
technique, which has proved useful for obtaining the goals for reengineering the current system.

- **Fourth Phase:** The addition of new actors in the system is based on those i* modelling techniques that create a first social i* model, and then insert the system actor in order to obtain the socio-technical one [5, 11, 21]. The addition of new actors and the reallocation of responsibilities is also addressed in [7, 31].

- **Fifth Phase:** The evaluation of alternatives using the REACT method has been described and validated in [13, 43]. Other proposals that include structural metrics are [7, 16].

- **Sixth Phase:** PR/M proposes the generation of the new information system specification from the i* model of the chosen alternative, which can be done applying the proposal in [42, 11].

We remark that there are many other methods and techniques that could have been used within the context of PR/M, either from the requirements engineering or business process reengineering fields. For instance, the first phase of PR/M captures and records information about the different elements of the current processes. To obtain this information alternative techniques can be applied, as long as these techniques fulfill the objectives of the phase, namely:

- Decompose the business processes in several activities, where each activity groups actions of the same nature.
- Describe activity properties, stating the actors involved in each activity (e.g., human, software and organization), its preconditions, postconditions and triggering events, and other additional assumptions and constraints.
- Describe, for each activity, its normal and alternative courses of actions, specifying the actors and resources involved.
- If the business processes currently include a software system, the system boundaries have to be clearly stated.

Thus, it is possible to exchange techniques used in each of the phases. To facilitate this exchange we have defined ReeF [25], a Reengineering Framework that uses the Method Engineering approach [6, 39] to define the generic set of phases that can be customized by other methods and techniques.

### 13. Experimentation

PR/M is an innovative method that has yet to be validated in industrial case studies. However, we have designed an experimentation plan that is organized in three stages using academic exemplars. Since our claim is that PR/M solves some of the open issues in existing i* methodologies, especially in terms of providing guidelines for constructing the models in a systematic way, the first two stages of PR/M use exemplars reported by other authors as a basis for comparison.

In the first stage we decided to look for the most common exemplar in goal- and agent-oriented proposals, namely the Meeting Scheduler case study, based on the Meeting Scheduler Problem Statement [33]. This problem has been addressed by many authors in the goal- and agent-oriented communities [10, 38] and was chosen by Yu to illustrate his i* proposal [48]. We presented the application of PR/m to the Meeting Scheduler case study in [19].

The application of the method over this small-scale problem helped us to check and improve the rules, guidelines, checks and techniques that we initially proposed within the method. Comparing our results with the solutions obtained by the other authors allowed us to validate aspects such as the adequacy of the construction of the i* model, as well as compare the analyzed and evaluated alternatives with the already-specified candidate solutions. In particular, results provided us with confidence in the rationale behind model construction, since we were able to trace back any single element appearing in the model of the initial process. We also applied the patterns for the generation of alternatives, checking that it is possible to define desired properties in terms of structural metrics.

In the second stage we looked for a medium-size exemplar for which some form of evaluation was available. We chose the eMedia Shop problem statement [32]. PR/M was applied to model the eMedia shop as a social system (Phases 1 and 2) to which we added the socio-technical system (Phases 3 and 4) following the strategy proposed in TROPOS [5]. As a result we obtained two different models, one for each system that presented the same underlying concepts than the ones identified in [32], especially when modelling goals, softgoals and some resources. As [32] also addresses the generation and evaluation of candidate organizational structures, we tried to compare the results obtained in both approaches. However, this has not been possible because of significant differences between the number and type of dependencies obtained by means of PR/M. One reason for this was that PR/M is very precise in defining the models and therefore more dependencies appear and thus, the organizational alternatives compared in [32], are not equivalent from the intentional point of view. Although one could argue that this fact aligns with the intrinsic freedom that i* provides, it was clear that comparison of organizational alternatives has to be based on models developed using the same principles.

The third stage is the consolidation of the model. We have chosen the eLearning System, an excerpt of which is reported in this paper. Since it is large-scale, one of the main goals has been how to tackle the complexity that i* models exhibit in these cases. The very nature of the PR/M method (guidelines, rules, checks, etc.), its definition of fine-grain units (we analyze business processes by decomposing the problem into use cases), and tool support to automate and guide cumbersome tasks, has delivered satisfactory results with an appropriate level of effort.
From the industrial point of view PR/M has benefited from lessons learned obtained with the RESCUE process, which may be considered a predecessor for its first phases. In the context of RESCUE, context modelling, human activity models and i* have been applied successfully to model and analyze requirements for new socio-technical systems in three air traffic management projects: for en-route conflict resolution (CORA-2), enhanced air space management (EASM), and departure management (DMAN). Detailed HAMs were developed in the DMAN project to describe current processes when managing departures from airports. In this section we report experiences from the DMAN project. For more information about the CORA-2 we refer to [34].

DMAN was Eurocontrol’s new system for scheduling and managing departures from major European airports. In the DMAN project context modelling, human activity models and i* were used to analyze both the current process models and the requirements specification for the future system. RESCUE applied verification checks to the models at different stages in the process. During these checks, the human activity models revealed that important human actions (typically physical and cognitive actions undertaken by the air traffic controller) were often omitted from the future system specification, which tended to specify system computation and interaction behaviour rather than the behaviour of human actors in the wider socio-technical system. This direct comparison of the current and future system models led to important revisions to the DMAN use case specifications. More details of these checks and their consequences are reported in [35].

One problem that emerged with RESCUE when developing an i* SD model is the establishment of all the possible dependencies. Analysts in the projects found it difficult to know how to ensure that all possible dependencies had been considered. The solution implemented in RESCUE consists of, before drawing a first-cut graphical SD model, listing possible dependencies between actor pairs in a tabular notation. As this has proven to be an effective practice, in PR/M we propose to write these tabular dependencies in a more systematic way by analyzing the different actions undertaken in the process and documenting them in the DIS template. As mentioned in [31], an effective way of doing that is to enumerate chronologically all of the actions that need to be executed until completing the activity, making explicit both the actions that the actor performs by itself and the actions that the actors requires from other actors.

We are currently preparing a more exhaustive long-term validation based on guidelines provided by Wohlin et al. [47]. We have identified two different populations to work with in different experiments – undergraduate students without any experience on i* and researchers and developers with interest or even experience on i*. For this second population we believe that the recently created collaborative wiki on i* [27] may play a crucial role.

14. Conclusions

Information systems are rarely developed from scratch. Rather they often automate existing social processes or socio-technical systems. We argue that information system development can be addressed as a reengineering process and, in that particular context, business process reengineering provides an adequate framework for information systems specification.

In many cases business process reengineering tends to be a complex activity due to the scale of current information systems. It has been argued that the complexity of the reengineering process forces people to invest too much effort into the as-is model without getting enough value out of it. This can be true in some approaches, but solving that issue has been one of the motivations of PR/M.

PR/M is a six-phase method which extracts a strategic i* model of the current process, analyses the resulting model for improvement, generates different alternatives, facilitates their evaluation and allows specification of the new system.

The analysis of the current processes provides a better understanding of how things are done and the rationale behind them. Also, during the reengineering process a more in-depth study of process alternatives allows analysts to detect strengths and weaknesses in the current process, which in turn aids them to foresee the impact of the modifications on human work. This last aspect is a key point in the industrial sector in which organizational systems are intended to satisfy and improve the organizational strategies.

We argue that PR/M reduces the initial effort required to model current processes whilst delivering real benefits through rules, guidelines, methods and checks that can be systematically applied for guiding the construction of the as-is model from the observation of the current process into the generation of the new system specification. We believe that this reduces the needed effort and increases the accuracy of the final specification.

Furthermore, PR/M is consistent with current trends in business process modelling that support the simulation, enactment, and analysis of processes. To simulate, recent proposals [9, 17] show how processes modelled in i* can be made operational by using a logical support or simulation languages. To enact, recent work has been presented to study how i* models can be translated into BPEL [44]. And to analyze, PR/M itself provides a framework for process analysis using structural metrics.

Among the key points of our proposal, we remark:

- It makes use of agent-oriented and goal-oriented concepts which allow expressing intentionality, beliefs, dependencies, etc. This kind of formalism supports strategic reasoning and therefore the models developed and analyzed are close to the abstraction level in which those decisions that imply the business processes of the organization are taken.
• It provides a well-defined method for system specification which is more prescriptive and provides a deeper level of detail than usual for those kind of comprehensive methods. In particular, we have shown in the related work section how PRM compares with other methods issued in the context of agent-orientation.
• It involves the use of consolidated requirement engineering techniques, putting together concepts from goal-oriented modellling (e.g., the i* language, KAOS patterns, organizational patterns), specification (e.g., human activity models, use case diagrams), knowledge gathering (e.g., inquiry cycle, scenarios) and others.

At the same time, we are proposing new techniques for those points of PRM in which we have not found adequate support in the current state of the art. Among them, we mention:

• The distinction among Operational and Intentional i* Models facilitates the construction of the departing model and makes the process more guided.
• The use of the KAOS goal classification and the ISO/IEC 9126-1 [26] quality attributes catalogue for driving goal and softgoal identification, for building the Intentional i* model.
• The proposal of different patterns of goal responsibility assignment and goal operationalization which allow exploring alternative solutions in a systematic way in order to represent feasible behaviours.
• The evaluation of those alternatives is done over a framework for defining goal-oriented metrics which analyse the properties of interest considering the shape of the analyzed model.
• The different alternatives considered for the new system are defined taking into account both functional and non-functional aspects, elaborated accordingly to the strategy of the organization. Alternatives can be compared if we check that there is Intentional Equivalence between them.

As we have commented in the experimentation section, PRM has not yet being validated in the industrial setting. However the preliminary results on academic case studies are encouraging. Concerning limitations of the approach, our major concern is with scalability, which is usually a problem in i*-related approaches. Our conjecture is that PRM may scale reasonably well because the method is so guided that at every moment, it is very clear which step to undertake, which patterns can be applied, which answers are adequate, etc. In the other hand, PRM allows processes to be analysed individually for the details of their achievement, i* models for individual business process are not too large.

The very nature of PRM has made possible the development of J-PRM [23], a tool that supports the whole method. The use of J-PRM is crucial to address scalability; in fact, we keep on developing the tool in order to adapt it to our needs when performing more case studies.

Acknowledgements

This work has been partially supported by the CICYT programme, project TIN2004-07461-C02-01. Gemma Grau work is supported by an UPC research scholarship. J-PRM has been developed by Sebastián Avila, Gemma Grau and Marc Oriol. The authors want to thank the REFSQ'2005 workshop organizers, anonymous reviewers and assistants for their comments and suggestions on earlier versions of this paper.

References


