Business Modeling in Process-Oriented Organizations for RUP-based Software Development

Francisco J. Duarte[1,3], João M. Fernandes[2], Ricardo J. Machado[3]
[1] Blaupunkt Auto-Rádio Portugal, Braga
[2] Dept. Informática, Universidade do Minho, Braga
[3] Dept. Sistemas de Informação, Universidade do Minho, Guimarães
PORTUGAL

Abstract

Several organizations are nowadays not particularly comfortable with their internal structuring based on a hierarchical arrangement (sub-divided in departments), where collaborators with a limited view of the overall organization perform their activities. Those organizations recognize the need to move to a model where multi-skilled teams run horizontal business processes that cross the organization, and impact suppliers and clients. Therefore, to develop software systems for those organizations, the development team must necessarily address two aspects: (1) the development process must be appropriate and controlled, and (2) the organizational platform where the organizational processes will run must be modeled and considered, both in the software development and the target organizations.

In this chapter, a proposal of a generic framework for process-oriented software houses is presented and the respective way of managing the process model and the instantiation of their processes with the Rational Unified Process (RUP) disciplines, whenever they are available, or with other kind of processes is recommended.

Additionaly, it is presented how the generic framework was executed in a real project called "Premium Wage" and shown, in some detail, the created artifacts (which include several UML models) during the development phases following the RUP disciplines, especially the artifacts produced for business modeling.

Keywords: RUP – Software development process – Process-oriented organizations – Business modeling.

1. Introduction

1.1 Process-Oriented Organizations

The concept of a process-oriented organization is a way of focusing the activities of an organization towards the clients needs [Hammer, 1996]. These activities are oriented towards and validated by the clients, whose necessities must be satisfied efficiently and

with quality. Reengineering, and its process-orientation, must be applied to anticipate change and not as a corrective procedure when bad business indicators occur.

In process-oriented organizations, clients' needs must be continuously satisfied, which mandates an easy and fast adaptation to changes. This favors and forces the continuous improvement of every aspect of the enterprise, being it process-, product-, or organizational-related.

Information technologies are among the principal factors to permit a process-based restructuring of a given organization [Spurr, 1994]. The development of a software application for organizations of this kind must consider their process framework. Thus, the software engineering processes must take into account the organization structure. With this model, the application becomes more useful to the target organization, and maintenance is facilitated since no major modifications and adaptations to the process framework are needed.

A process framework inside an organization contains processes, and these can be viewed as a set of activities that has as inputs a set of services and/or materials and has as outputs also a set of services and materials. This view must be oriented towards the necessities of the client and to the creation of added-value. This implies that the clients' requirements must always be considered, both in the design and in the performance of the system.

In an organization, there are other processes rather than those that provide added-value to the clients. The existence of different types of processes is necessary to assure, for example, the strategic planning for the organization, the recruitment of the human resources, or the fiscal duties. As illustrated in fig. 1, these processes are instantiated in Management and Support Processes.

The generic process framework presented in [Fernandes & Duarte, 2004] was customized to the specific case of organizations that develop software (software houses) and next we describe its main characteristics.

Within an organization, the management by processes requires a structure that differs from the typical functional hierarchy. It is mandatory to synchronize the processes among them and to fulfill the strategic objectives of the organization. For a process-oriented organization, a structure with the following elements should exist:

Process Management Top Team: This team includes the top managers, all process owners, and, if exists, the process management structural responsible. Its mission is to revise all the processes according to the strategic objectives of the organization, to analyze the effectiveness of the process-oriented management, and to decide about unsolved problems at the processes' interfaces.

Process Sponsor: The mission of this top manager is to help and instruct the process owner, to decide when there is a problem of interface among processes, to determine the strategic orientation of the process, and to assure that the process is uniform within the organization.

Process Owner: For each process, its owner must have know-how on managing processes and persons and competency in the areas associated with the process. His mission is to lead the process' multi-disciplinary team.

Multi-Disciplinary Team: This team must be created for each added-value process, since they represent the most important processes for the clients. The mission of this team

is multi-fold: to monitor its process, to define and analyze the key indicators and the process objectives, to ensure that the process documentation is updated, to decide when and how to use improvement teams and to coordinate them, and to manage the process execution teams.

Execution Teams and Team Leaders: These teams and their leaders represent the instances of a given process [Scheer & Nüttgens, 2000]. Therefore, during the execution of a process, some teams will use it with a specific focus. For example, for a given production process, one team may be responsible for producing parts for industrial clients, while other team may produce them for individual clients.

To align a process-based organization with its strategic objectives, it is crucial that the goals are based on the organization mission and vision, and also on its principles and values. Based on those strategic objectives and in the business plan, the priority when deciding the key business processes within the organization can be perceived and included in the process landscape. This action may imply that some processes, activities, or tasks can be eliminated if they do not add any value to the clients, neither to the organization. These eliminated (or redefined) processes, activities, and tasks and their respective consequences in terms of reorganization and impact in human resources are the essence of re-engineering [Hammer, 1996].

1.2 Case Study

The proposals made in this paper were tested in a real industrial environment, more specifically in the first author's organization.

The project, entitled "Premium Wage", consists on the development of a software application to calculate the payment of extra money to employees, based on their productivity, quality, and absenteeism [Fernandes & Duarte, 2004]. This project was considered critical, since it is likely to have important social and behavioral impacts on the organization, namely if the amount is badly calculated or if it is impossible to explain how it was obtained. This premium was introduced with the aim of ameliorating the organization's overall productivity and quality, and to return the excellence to the workers.

Besides its criticality, the business process is also complex since it depends on other processes. In this case, the payment of a premium depends on three main factors: individual absenteeism, quality of the products made in the employee's line, and individual performance. The first two sub-processes were extended in order to support new functionalities. For the third, a complete reengineering was carried on. Finally, to the premium wage calculation a new process was designed, modeled, and implemented.

In the project, the proposed framework, namely RUP's business modeling discipline is extensively used and we evaluate the capacity of the process to cope with complex organizations.

1.3 Structure of the paper

This paper is structured in 5 main chapters. Chapter 2 presents the main characteristics of the framework for Process-Oriented software houses, namely the processes it is composed of. In chapter 3, we describe RUP's business modeling core discipline, which implements an added-value process. Chapter 4 describes and discusses in detail the produced artifacts for the case study during the execution of the business modeling discipline.
In chapter 5, future trends and work along with the conclusions are presented.

2. Framework for Process-Oriented Software Houses

A reference framework, also called PSEE (Process-centered Software Engineering Environment), does not support the notion of a predefined process model that is supposed to be applied in every development project, but instead supports a wider variety of processes based on parameterization [Engels *et al.*, 2001]. SPADE [Bandinelli *et al.*, 1994], EPOS [Conradi *et al.*, 1994], MELMAC [Gruhn & Jegelka, 1992], OIKOS [Montangero & Ambriola, 1994], OPEN [Henderson-Sellers, 2000], and APPL/A [Sutton *et al.*, 1995] are well-known examples of PSEEs systems or process modeling formalisms.
A reference framework for process-oriented organizations is presented and justified in [Fernandes & Duarte, 2005]. In this chapter, an updated version of this particular framework (fig. 1) includes a new process, called *change management* that allows a more explicit management of continuous improvement and changes. This framework has some of its processes consubstantiated with RUP's disciplines. Since RUP is a process meta-model, we can obtain a specific process model by tuning some of the parameters, allowing our reference framework to be tailored for each kind of project.

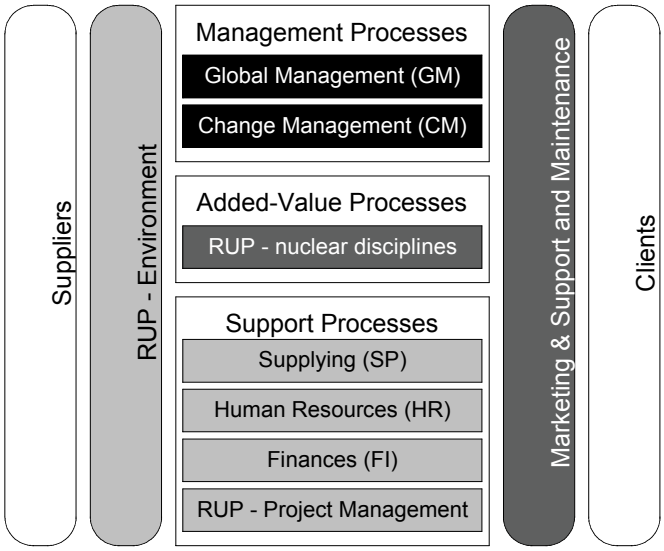

Fig. 1: Reference framework for software houses.

In software houses, we propose the business processes to be organized into two different groups: (1) the first one includes processes that exist in any organization independently of its relation to the software development business; (2) the second group includes processes that present specific characteristics due to the fact that the organization main activity is the development of software-based solutions.

In the first group, we can observe the following processes:

- Any type of organization needs a *global management* process, which includes the sub-processes *global strategy* (GS), *policy deployment* (PD), and *business plan* (BP). This process is equivalent to that of any other organization, although we must take into account the particularities of the software market, such as the rapid changes in technology and the competition in worldwide markets when defining, for instance, organizations' business and vision.
- Once stable processes models have been obtained, they should be released and afterwards it is desirable to manage its change, according to the change requests made by the process stakeholders [Gruhn and Wellen, 2000]. The *change management* (CM) process allows the software house to collect, organize, and manage the output data and experiences, that are the basis for processes self improvement. This new process should exist in any software house aiming to reach the highest CMM (*Capability Maturity Model*) levels [Paulk *et al.*, 1995]. To reach the highest CMM levels, the concern about the constant improvement of the development processes must be part of each process, instead of being only a matter of the CM process. CMM level 3, which is considered the minimum when discussing about the software process [Henderson-Sellers, 2000] is reachable with RUP if it is extended accordingly, for example, to the proposals made in [Manzoni and Price, 2003]. Also, rules and procedures should be defined previously to cope with the introduction of significant changes in the organization.
- The *supplying* (SP) process consists essentially in creating copies of an application. In contrast with more traditional industries, where it represents probably the most important process, in software, due again to its intangible nature, this is a trivial process. The usual outsourcing of this process comes from the fact that it is considered to be secondary for an organization that develops software. Therefore in this kind of organization, this process is a support one.
- The *human resources* (HR) process for software factories is the same as for other types of organizations. We must however point out that software development requires highly specialized people, being their hiring a critical issue for the success of the organization. It is impossible to produce quality software without skilled people.
- The *finances* (FI) process is the typical fulfillment of the fiscal obligations, which is common to all types of organizations and may also include controlling activities.
- The process *marketing*, and *maintenance and support* represent the typical *customer relationship management* (CRM) process. This ensures that, when a

software application is delivered to the final clients, its life-cycle does not end by at that time, but instead continues with this process, incorporating changes and corrections, providing training to the users, while the application is being used by the clients. Included, as a sub-process, can also exist hotline support activities.

In the second group, the suggested processes are influenced by the fact that RUP constitutes a systematic approach to assign tasks and responsibilities to its members. The main aim of RUP is to construct quality software that meets the requirements of the stakeholders, within a typical engineering context [Machado *et al*., 2005]. RUP identifies and defines the activities needed to map user requirements into a software application and is accepted to be a generic/customizable process that can be adapted for a wide range of contexts, namely organizations with distinct CMM levels, different skills and tools, and unequal number of team members. Thus, the second group of processes includes the following RUP-based processes:

- Since software is an intangible product, it is obvious that no raw materials are needed to produce it. For organizations that develop software, RUP's *environment* discipline can instantiate the *supplier relationship management* (SRM) process, since it furnishes the working environment (*e.g.* development tools) and the development guidelines to be followed by the teams.
- The RUP's core disciplines (*business modeling, requirements, analysis and design, implementation, test, and deployment*) represent the most critical activities for an organization that develops software and can be seen as the *time-to-market* (TTM) process of the organization. This set of activities, or sub-processes, run in parallel for the same development project.
- The RUP's discipline *project management* implements the *data management* (DM) process. In this discipline, some activities lead to the production of indicators of the project status. Its existence is the foundation to take decisions based on facts, related to the advance of the project aiming to adjust and improve the software development process.

Our framework intends to cope with all issues related to software development processes and potentially can be used with the following purposes: documentation, analysis and improvement of software process models [Bandinelli *et al.*, 1993], software process improvement [Avrilionis *et al.*, 1996] and software process execution [Deiters & Gruhn, 1998].

Fig. 1 represents a top-level view of the process landscape, which is useful to show and discuss with top-managers and process owners. Afterwards, this view must be further refined by the process owner and the multidisciplinary team, to present the process at the appropriate level of detail for each software house professional.

It is also important to notice that fig. 1 corresponds to a specialization of the general framework presented in [Fernandes & Duarte, 2005] for the particular situation of organizations whose main activity is to develop software. The execution of the TTM process by the software house produces an instance of the general framework that models the organization where the software will run. This fact makes the general framework

valuable in two senses: (i) as a reference for the software house to model itself, and, (ii) as a reference for the software house to model the target organization.

3. Business Modeling in RUP

RUP's core disciplines implement added-value process. These disciplines are sub-divided in activities, which can be viewed as sub-processes. The description of those sub-processes is made with UML activity diagrams, complemented optionally with other kinds of diagrams, such as interaction and business object diagrams.
This representation is also valid for all other processes of a generic organization. At any time a software house starts a new project, the TTM process is executed. Since we propose this process to be implemented by the six RUP's core disciplines, it implies that *business modeling* will also be executed. Among the recommended diagrams by this discipline for modeling purposes are included activity diagrams. Thus, a target organization will be modeled also by a collection of these diagrams. Additionally, within the software house, the discipline *business modeling* itself can also be modeled by activity diagrams, since it is a sub-process of the TTM process (fig. 2).

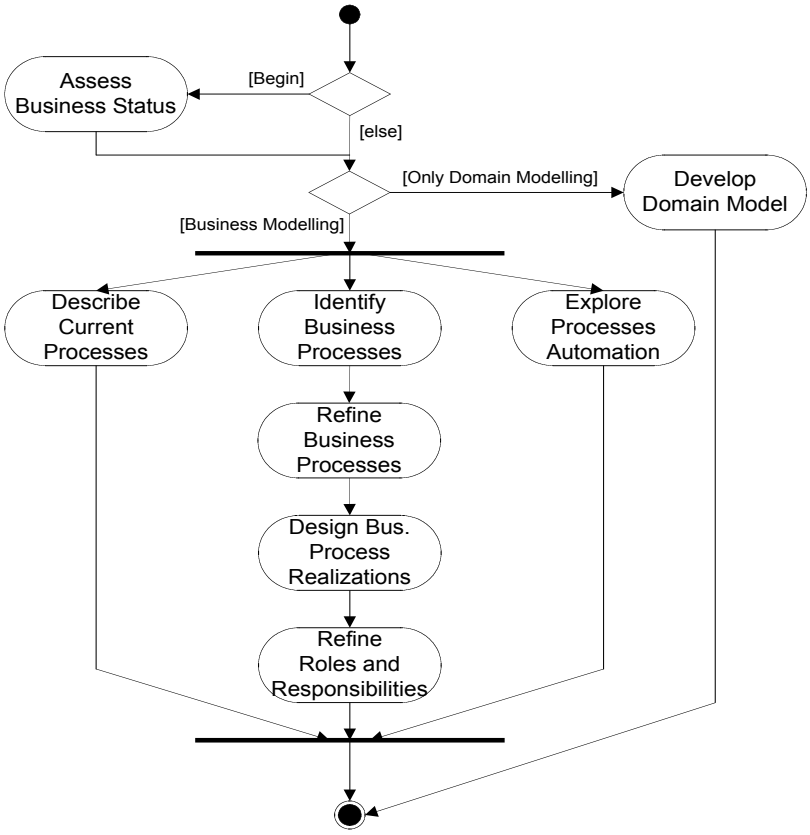

Fig. 2: Activity diagram to help the execution of RUP's business modeling discipline in TTM process.

During software development, all the stakeholders must have a common understanding of the business processes that exist in the target organization. This reality is not circumscribed to the obvious organizational information systems.
If the development of applications does not take into account the current business processes (or those to be implemented), the result will be probably unsuccessful. This may be caused by the fact that end users do not employ correctly the application, since it does not support directly the activities under their responsibility.
The main activities of *business modeling* are centered on the identification, refinement, and realization of the business processes and in the definition of the roles of people associated to the business. Each role in this RUP's discipline has under its responsibility the execution of several activities that will have as deliverables several artifacts (table 1).

Role	Sub-Activity	Activity (fig. 2)	Main Artifacts
Analyst of the Business Process	Assess target organization	Assess Business Status	Business rules Business use case model Business Glossary Business object model Business vision Supplem. business specification Target organization verification Business architecture
	Set and adjust goals		
	Capture the business vocabulary		
	Find business actors and use cases	Describe Current Processes	
	Maintain the business rules	Assess Business Status & Identify business processes	
	Structure business use case model	Refine Business Processes	
	Define the business architecture	Identify Business Processes	
Reviewer of the Business Model	Review the business use case model	Refine Business Processes	
	Review the business object model	Refine Roles and Responsibilities	
Designer of the Business	Detail business use cases	Refine Business Processes	Organizational units
	Find business workers and entities	Describe Current Processes	
	Define the automation requirements	Explore Processes Automation	
	Detail business entities	Refine Roles and Responsibilities	
	Detail business workers		

Table 1: Roles, activities, and main artifacts for business modeling in RUP.

The activities of table 1 are at a detailed level than those of fig 2. For example, the activity *refine business processes* includes the activities *structure the business use case model*, *review the business use case model*, and *detail business use cases*.
Among all the activities and their respective artifacts, only some are mandatory. This flexibility permits the configuration of RUP, so that it can be adapted to a specific project executed in a given organization.

4. Business Artifacts for the Case Study

Nowadays, the technology-planning horizon for big companies is a synthesis of software engineering and process engineering [Smith & Fingar, 2002]. Additionally, the success of a project depends heavily on the correct perception of the business process to be modeled. Taking into account these two aspects, the RUP's *business modeling* discipline assumes a critical role in the software development process. The artifacts that can be generated by this discipline have the following objectives:

- To understand the structure and dynamics of the organization where the system will be executed;
- To comprehend the current problems of the target organization and to identify potential improvements;
- To assure that clients, final users, and developers have a common understanding about the target organization;
- To capture/deduct the requirements of the system necessary to support the target organization.

RUP can be parameterized and used both in small and complex projects and, next we discuss what artifacts were produced for the case study. For this parameterization to occur it is necessary, during project execution, to choose which artifacts to use and their level of detail. This choice was validated by quality assessments contained in the process model as milestones between phase transitions. Thus, both the subset of used artifacts and also its degree of detail can not be anticipated with rigor, but must be selected based on experience and knowledge of the development team in relating the characteristics of each project with the functionalities offered by the artifacts. The criteria to fulfill this choice are related with:

- Characteristics of the project itself (e.g. criticality of the modeled business processes, type of target organization);
- Characteristics of the organization that develops software (e.g. team size, level of knowledge about internal rules);
- Temporal restrictions. Since resources are limited in engineering projects, it is always necessary a balance between the quantity and detail of the produced artifacts and the deadlines for implementing the project.

The produced artifacts result from a set of activities that occur inside those disciplines. In this case study, we identified the need for the artifacts to represent two distinct situations in terms of business: one part of the project represents reengineering activities of some business processes, while the other part represents the introduction of a new business process.
In several diagrams (e.g. Business Use Case Model), the standard UML is augmented with the stereotypes defined by RUP, thus allowing the creation of RUP-like artifacts.

4.1 Business Rules

The business rules correspond to policy statements and conditions that should be fulfilled, from the business perspective. They are similar to systems requirements, but

they focus on the business core, expressing rules related to business, and also its architecture and style. Its modeling must be rigorous, being one possibility the usage of the Object Constraint Language (OCL) as specified in UML. Alternatively, business rules can be modeled with structured English, using some fixed constructors [Odell, 1998].
The usage of structured natural language with fixed constructs and with a pseudo-programming language syntax was chosen due to the necessity to validate directly with key users the perception of the development team about stated business rules, and also because key users have a generic engineering background. This way, in the particular situation of the case study, the usage of natural language was preferred over OCL.
Table 2 shows, as examples, three rules for Premium Wage system.

Rule #	Description
1	**If** worker_has_conflict_inside_team **or** worker_is_under_disciplinary_process **then** premium(worker) = 0
2	**Switch** absenteeism(worker) **case** = 0h: premium(worker) = premium(worker) * 1 **case**]0-4h]: premium(worker) = premium(worker) * 0.75 **case**]4-8h]: premium(worker) = premium(worker) * 0.5 **case** >8h: premium(worker) = premium(worker) * 0
3	**If** line_productivity_not_available **or** absenteeism_not_available **or** quality_factor_not_available **then** premium(worker) = 0

Table 2: Business rules examples for Premium Wage.

Rule 1 describes a situation where no premium payment is due when the worker has conflicts in its team or is under a disciplinary process, even though productivity, quality, and absenteeism are at good level for that worker. Rule 2 assigns a weight factor to the final premium based on the worker's absenteeism. Rule 3 states that if there is no information available from one of the three premium factors, no payment will be made for that worker.

4.2 Business Use Case Model

The main goal of this artifact is to show how the business is being perceived and run by stakeholders. This is achieved by modeling the business processes and their interactions with external parties, based on business use case diagrams (with stereotypes for business use cases and business actors) [Fernandes & Machado, 2001], [Machado & Fernandes, 2002].
Business processes model should specify how added value is created for the business actors. Activity diagrams, possibly extended with the representation of organizational units interfering in the business process and with the distribution of the activities by those organizational units, can support this modeling. The knowledge about 'who is doing what' should be obvious when reading this model.

Business use case model is the first description of the business functionalities and actors inside the target organization. For Premium Wage one artifact was created to model the current situation (fig. 3) and another for the desired future situation (fig. 4). The existence of these two models shows to all stakeholders, using the same notation and same detail level, the first perspective of effort amount needed to reach future situation, and more important, acts as a base for target organization's management decide on business re-engineering and improvement. A special emphasis should be give to explain to target organization's management that an information system is not the complete business reality, but an abstraction of it. Real business processes, such the ones stated in *business modeling* discipline are far more extent than the ones implemented inside the information system. This may be caused by people's activities because not everyday they act equal, and mainly because people like to have their own special information systems (e.g. spreadsheets, or personal databases) to make decisions and causing the *starvation* of the organization business information system with missing data. This is crucial success factor at present time, because the work is no longer only individual or departmental related, but exists through horizontal business processes that cross the entire organization and reaches external partners.

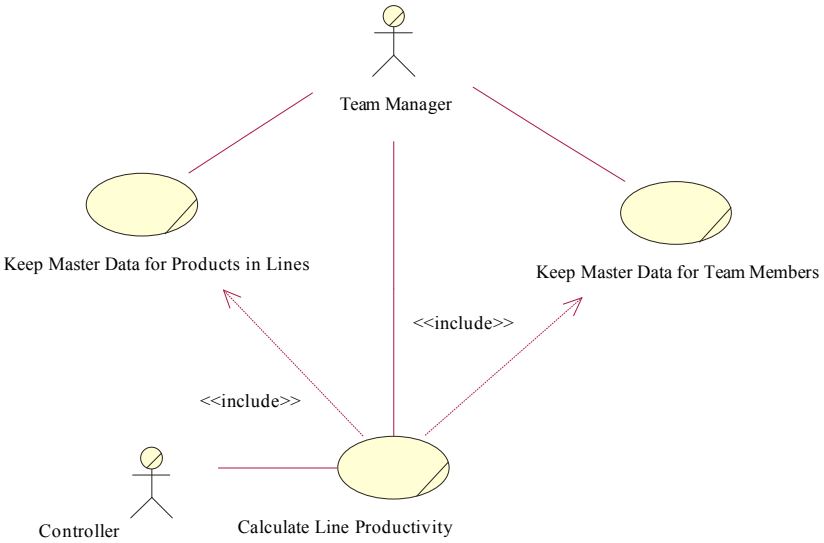

Fig. 3: Business use case model – As-Is situation.

Fig 3 shows the current situation, where only three business use cases exist and only two business actors takes part. The line productivity is calculated by the controller based master data lines' team member and products maintained by the team manager. This situation shows that no information was available on how and where to collect quality data and absenteeism. Next, on fig. 4, the desired extensions and new business use cases can be seen. For the future situation six business actors are needed, and special care should be put on this situation, because some of them may not be directly related with project activities and may not understand the business value of the new

activities (stated in the business use cases) if no proper involvement and education is provided.

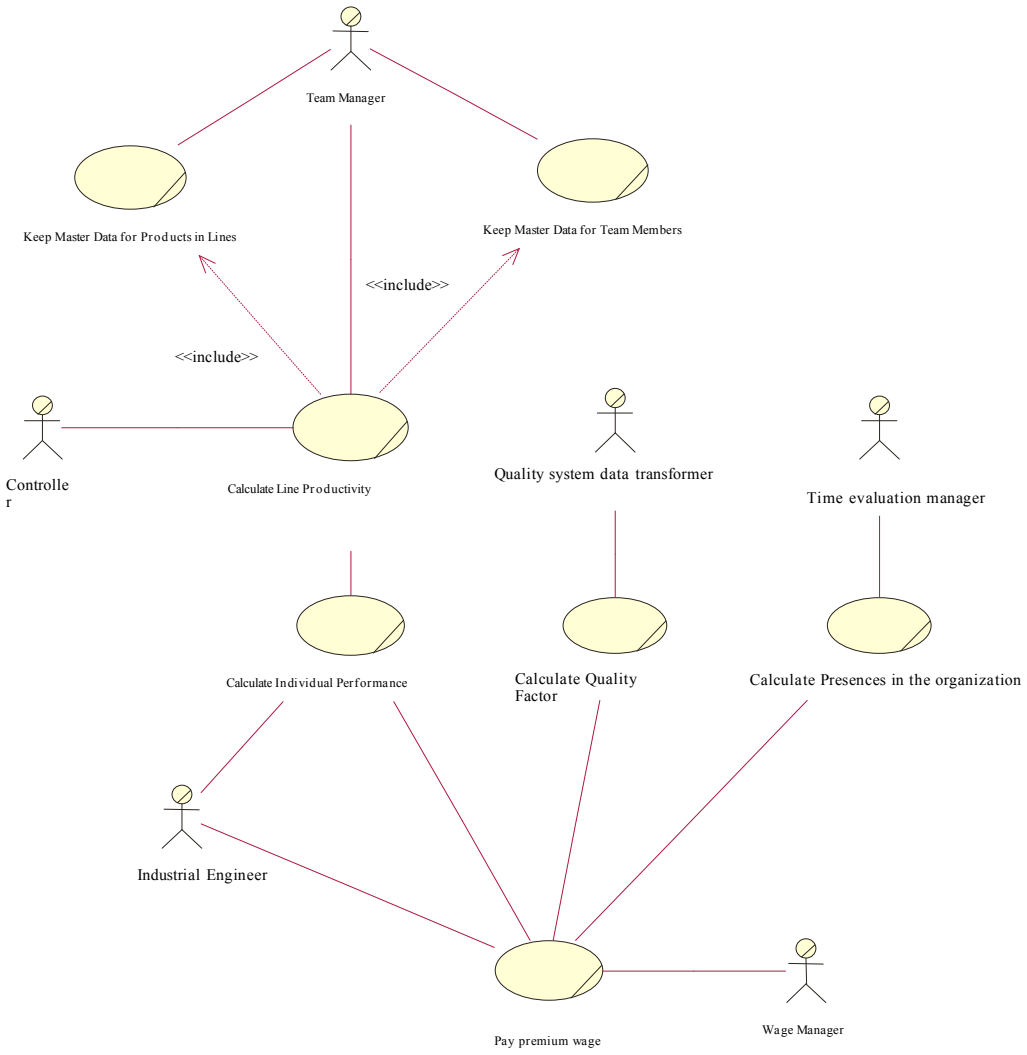

Fig. 4: Business use case model – To-Be situation.

In this figure situation, line productivity is *triggered* also by the controller, but to be useful, first has to be transformed into individual performance (because one employee can work in several production lines, for the premium calculation time frame). Additionally, the remaining two premium factors (quality and absenteeism) are stated, thus allowing the industrial engineer to calculate the final values and the wage manager to handle them.

4.3 Organizational Units

This artifact is used to reduce the complexity and structure the business object model by dividing it into smaller parts. For the case study, five organization units where created (fig. 5), each one representing a collection of business workers, business entities, relationships, business use-case realizations, diagrams, and other organization units.

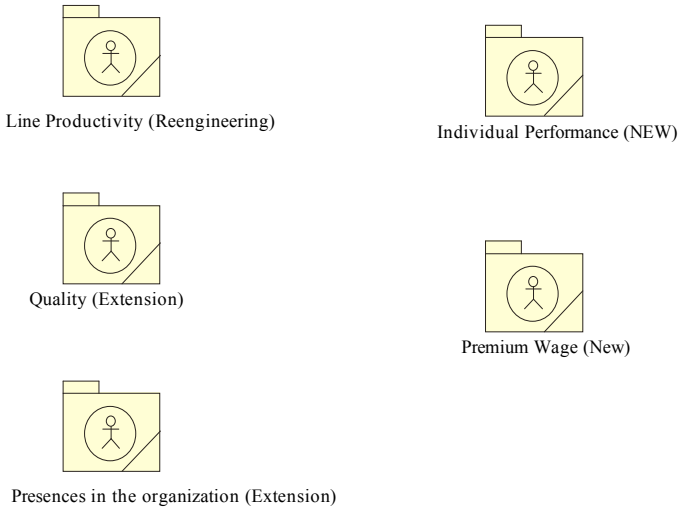

Fig. 5: Organizational Units.

The organization unit *line productivity* includes the re-engineered activities for the productivity calculation, *quality* and *presences in the organization* where extended to cope with premium calculation. The *individual performance* and *premium wage* calculation organizational units where newly created in the organization, because none of the existent could handle it.

4.4 Business Object Model

This artefact is an object model describing the realization of business use cases. It serves as an abstraction of how business workers and business entities need to be related and how they need to collaborate in order to perform the business. In fig. 6 and fig. 7 the realizations of some To-Be situation business use cases are presented, namely *Presences in the organization* and *Pay premium wage*.

The importance of this artefact comes also from being it a basis for identifying future information system actors and use cases, and in addition to identify classes for analysis and design models. First modelling the business and afterwards the information system is a crucial practice to correct align the business reality with the abstraction the information system implements. If the software information system development only starts with the modelling of software system, a quality software product can be obtained (being the quality perceived as the degree of correctly implement requirements) but that may fail completely to cope with business reality of the organization in which it will run. Although business information systems implement a large percentage of the

business practices, there are always some parts of the organization business processes there are not implemented in software systems. The modelling of such business practices by the software development team is a wise and safe procedure to avoid future inconsistencies when introducing the information system in the organization.

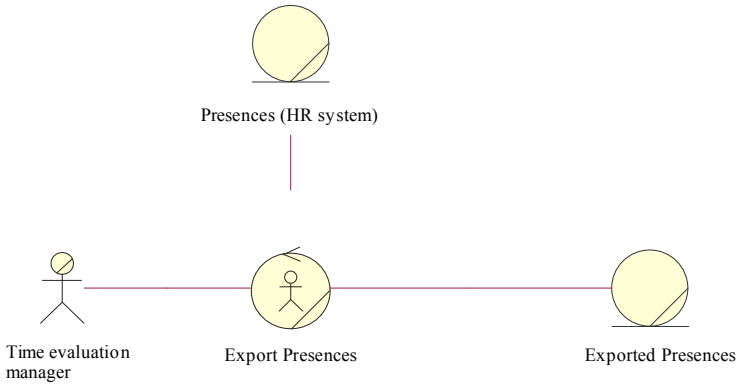

Fig. 6: Business object model - Calculate presences in the organization.

To calculate the presences in the organization (fig. 6), the business worker *Time evaluation manager* uses business entity *Export presences* to pick the business object *Presences* available in the HR system and generate a new business object *Exported presences* containing the presences in the organization in a proper format to use inside Premium Wage information system.
In fig.7, business entity *Calculate Premium Values* uses several business objects representing final premium factors , such as *Compiled Quality System data*, *Exported presences* (generated in fig.6 business object model), and *Individual Performance*, and also business objects representing levels and limits (acting as data supports to implement business rules (table 2), to generate the business object *Premium to Pay*. Afterwards, this business object will be used and transformed by several business entities and with and without the intervention of business actors, until the final *Employee Total Wage (Wage + Premium)* is loaded and calculated in the HR system, as shown in fig. 7.

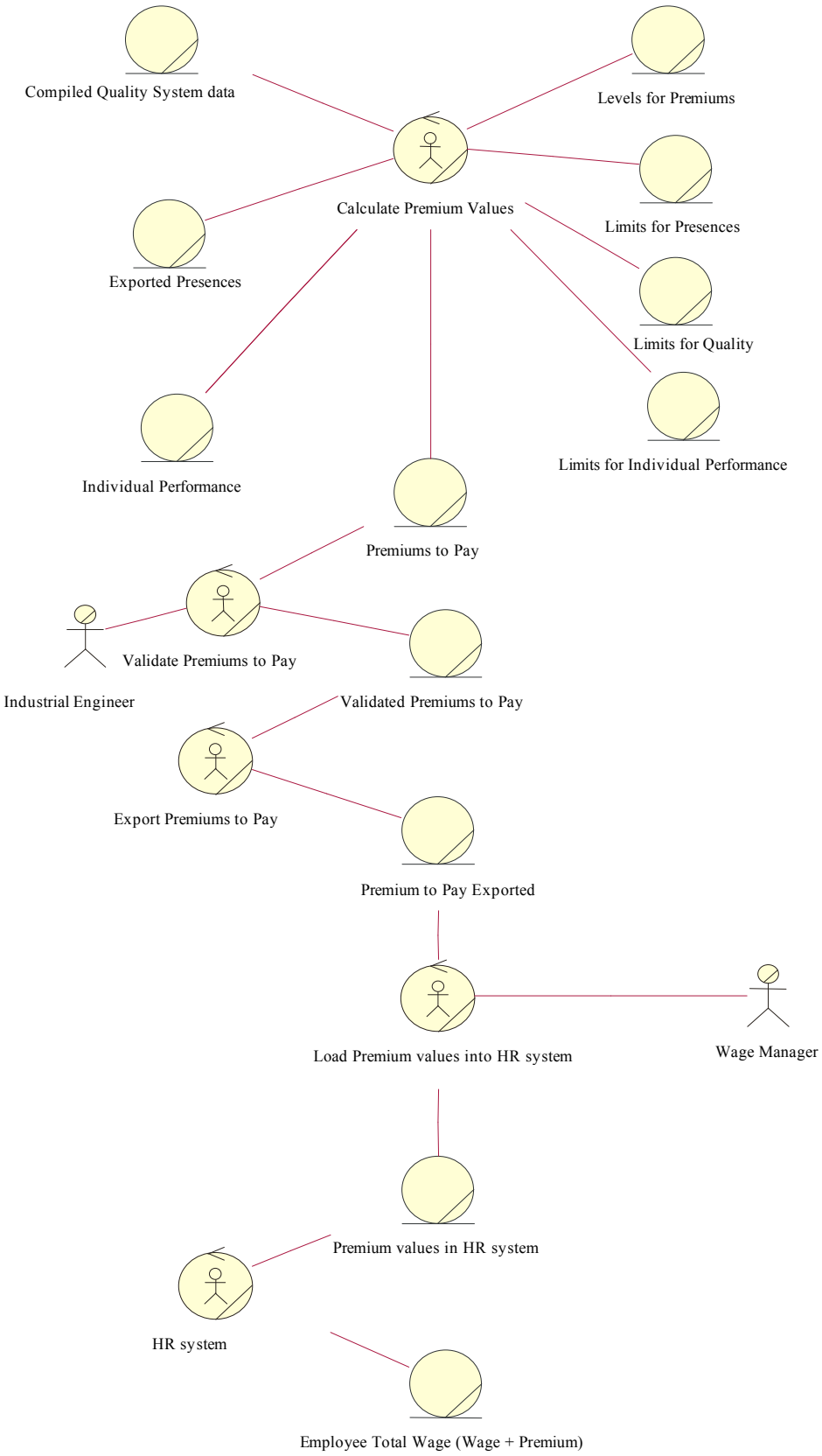

Fig. 7: Business object model - Pay premium wage.

The business object model exposed in fig. 7 is not fully contained in Premium Wage information system. The business object *Premium to Pay Exported* is generated by Premium Wage to be used inside the organization‘s HR system. This is an example where some of the business actors, business entities, and business objects will not be part of the information system required to the software development team, but they were modeled to guarantee a proper integration of current information systems (e.g. HR system) with newly developed ones (e.g. Premium Wage).

4.5 Other Business Artifacts

From the all proposed artifact in RUP for *business modeling*, some where not used in the Premium Wage case study. Next, we present the reason why they were not needed in the instance of RUP we used for this project:

Business Glossary: In this artifact all business terms and expression are kept. They are necessary for a good understanding among all project stakeholders. In our situation, the business terms are common to the target and to the software developing organization, because they both are sub-organizations of the same organization.

Business Vision: This artifact captures the goals of a particular business modeling activity, stating what is to be modeled and the reasons for it. It also serves as an oracle to all future high-level decisions. We did not used business vision because it is common to the target and software development organization due to the same reason of business glossary.

Supplementary Business Specification: No need for extra-detail than the one available in the Business Use Case and the Business Object Models.

Target Organization Verification: The target organization is perfectly known by the software developers. The current processes are modeled by the Business Use Case Model – As-Is Situation.

Business Architecture Document: The detail level presented in the Business Use Case Model and in the Business Object Model is sufficient to understand the business architecture;

5. Conclusions and Future Trends

In this paper, we have presented a revised version of a reference framework for process-oriented software houses, which serves as a foundation to model organizations. This specific framework is based on a more generic one, which is also used as a template to model the target organization in which the software product is to be executed.
Additionally, we also show the way of managing the framework, and the instantiation of its processes with RUP disciplines whenever feasible.

We also discuss in detail the usage of the framework within a case study, and more particularly, the produced artifacts during the execution of the RUPâ€™s business modeling discipline. The modeling capabilities of a graphical modeling language (such as the UML) and the understanding that it gives to all the stakeholders was a crucial factor in the case study to avoid communication and interpretation errors and to improve the solution utility and correctness.

As future work, we plan to formalize the processes of the reference framework, by using colored Petri nets [Jensen, 1992]. Similar approaches also based on Petri nets were also experienced with results [Gruhn & Wellen, 2001], [Aalst, 2003].
By adopting a formal language, it is possible to model, animate, simulate, and formally verify the properties of each single process. Additionally we intend to explicitly model the interfaces between the business processes in our framework, which allows the complete reference framework to be analyzed, verified, and validated.
We intend to automatically generate CPN skeletons from business requirements models. A semantic layer in the Arena environment [Kelton *et al.*, 2002], capable of accepting CPN based business specifications, will also be developed to allow the stochastic execution of workflow scenarios as a complement to some current validation approaches based on CPN/Tools [Beaudouin-Lafon *et al.*, 2001].
After formally describing the reference framework processes, we can use it in every organization (in this case, software houses) to compare with its current processes. This comparison (based on the same Petri net formalism) should allow a quick assessment of the organization against world-class processes, and consequently permit the re-engineering and improvement of its own processes. This way, the reference framework acts as a To-Be model to be compared with the As-Is model of the software house. The detected mismatches show the improvement areas for the software house to proceed accordingly within the organization vision and mission.

References

Aalst, W.M.P. van der (2003). Challenges in Business Process Management: verification of business processes using Petri nets. *Bulletin of the European Association for Theoretical Computer Science*, 80, 174-198.

Avrilionis, D., Belkhatir, N., & Cunin, P.-Y. (1996). Improving Software Process Modelling and Enactment Techniques. Montangero, C. (ed.), *5th European Workshop on Software Process Technology*, Nancy, France, Lecture Notes in Computer Science, 1149, 65–74.

Bandinelli, S., Fugetta, A., & Grigolli, A. (1993). Process Modelling in-the Large with SLANG. *2nd International Conference on the Software Process – Continuous Software Improvement*, Berlin, Germany, 75–83.

Bandinelli, S., Fuggetta, A., Ghezzi, C., Lavazza, L. (1994). SPADE: An environment for software process analysis, design, and enactment. Finkelstein, A., Kramer, J.,

Nuseibeh, B. (eds.), *Software Process Modeling and Technology*, Research Studies Press, London, U.K.

Beaudouin-Lafon, M., Mackay, W.E., Andersen, P., Janecek, P., Jensen, M., Lassen, M., Lund, K., Mortensen, K., Munck, S., Ratzer, A., Ravn, K., Christensen, S., Jensen, K. (2001). CPN/Tools: A Post-WIMP Interface for Editing and Simulating Coloured Petri Nets, *Application and Theory of Petri Nets 2001*, Proceedings of the 22nd International Conference, ICATPN 2001 Newcastle upon Tyne, UK.

Conradi, R., Hasgaseth, M., Larsen, J.-O., Nguyên, M.N., Munch, B.P., Westby, P.H., Zhu, W., Jacchert, M.L., Liu, C. (1994). EPOS: Object-oriented cooperative process modeling. Finkelstein, A., Kramer, J., Nuseibeh, B. (eds.), *Software Process Modeling and Technology*, Research Studies Press, London, U.K.

Deiters, W., Gruhn, V. (1998). Process Management in Practice - Applying the FUNSOFT Net Approach to Large Scale Processes. *Automated Software Engineering*, 5, 7–25.

Engels, G., Schäfer, W., Balzer, R., Gruhn, V. (2001). Process-centered software engineering environments: academic and industrial perspectives, *23rd International Conference on Software Engineering*, Toronto, Canada, IEEE CS Press, 671–673.

Fernandes, J.M., Duarte, F.J. (2005). A Reference Framework for Process-Oriented Software Development Organizations, *Software and Systems Modeling (SoSyM)*, Springer-Verlag, vol. 4, nr. 1, 94–105. http://dx.doi.org/10.1007/s10270-004-0063-0.

Fernandes, J.M., Duarte, F.J. (2004). Using RUP for Process-Oriented Organisations. Bomarius, F., Iida, H. (eds.), *5th Int. Conf. on Product Focused Software Process Improvement (PROFES 2004)*, Lecture Notes in Computer Science 3009, Springer-Verlag, 348–362.

Fernandes, J.M., Machado, R.J. (2001). From Use Cases to Objects: An Industrial Information Systems Case Study Analysis. *7th Int. Conf. on Object-Oriented Information Systems (OOIS '01)*, Springer-Verlag, 319–328.

Gruhn, V., Jegelka, R. (1992). An evaluation of FUNSOFT nets. *2nd European Workshop on Software Process Technology (EWSPT '92)*. Lecture Notes in Computer Science. Springer-Verlag, New York.

Gruhn, V., Wellen, U. (2000). Structuring Complex Software Processes by "Process Land-scaping", *7th European Workshop on Software Process Technology (EWSPT 2000)*, Kaprun, Austria, Lecture Notes in Computer Science, vol. 1780, Springer–Verlag, 138–149.

Gruhn, V., Wellen, U. (2001). Process Landscaping: Modelling Distributed Processes and Proving Properties of Distributed Process Models. *Unifying Petri Nets*, Lecture Notes in Computer Science; vol. 2128, Springer-Verlag, 103–125.

Hammer, M. (1996). Beyond Reengineering: How the Process-Centered Organization Is Changing Our Work and Our Lives. *Harper Collins*.

Henderson-Sellers, B. (2000). The OPEN Framework for Enhancing Productivity. *IEEE Software*, 17(2),53–58.

Jensen, K. (1992). Coloured Petri Nets - Basic Concepts, Analysis Methods and Practical Use, Vol. 1: Basic Concepts. *EATCS Monographs in Theoretical Computer Science*, Springer-Verlag.

Kelton, W.D., Sadowski, R.P., Sadowski, D.A. (2002). Simulation With ARENA, Second Edition, McGraw-Hill.

Machado, R.J., Fernandes, J.M. (2002). Heterogeneous Information Systems Integration: Organizations and Methodologies. Oivo M., Sirviö, S.K. (eds.), *Proceedings of the 4th International Conference on Product Focused Software Process Improvement – PROFES 02*, Rovaniemi, Finland, 629-643, Lecture Notes in Computer Science Series vol. 2559, Springer-Verlag.

Machado, R.J., Ramos, I., Fernandes, J.M. (2005). Specification of Requirements Models. Aurum, A., Wohlin, C. (eds.), *Engineering and Managing Software Requirements*, Springer-Verlag, 47-68.

Manzoni, L.V., Price, R.T. (2003). Identifying Extensions Required by RUP (Rational Unified Process) to Comply with CMM (Capability Maturity Model) Levels 2 and 3. *IEEE Transactions on Software Engineering*, 29(2), 181–192.

Montangero, C., Ambriola, V. (1994). OIKOS: Constructing process-centered SDEs. Finkelstein, A., Kramer, J., Nuseibeh, B. (eds.), *Software Process Modeling and Technology,* Research Studies Press, London, U.K.

Odell, J. (1998). Advanced Object-Oriented Analysis & Design Using UML. Cambridge University Press.

Paulk, M.C., Weber, C.V., Curtis, B., Chrissis, M.B., (eds.) (1995). The Capability Maturity Model: Guidelines for Improving the Software Process, Addison-Wesley, Reading, USA.

Scheer, A.W., Nüttgens, M. (2000). ARIS Architecture and Reference Models for Business Process Management. Aalst, W. van der, Desel, J., Oberweis, A. (eds.), *Business Process Management, Models, Techniques, and Empirical Studies*, Lecture Notes in Computer Science 1806, Springer, 376–89.

Smith, H., Fingar, P. (2002). Business Process Management: The Third Wave, Meghan-Kiffer Press.

Spurr, K., Layzell, P., Jennison, L., Richards, N. (1994). Software Assistance for Business Re-Engineering, John Wiley & Sons.

Sutton, S., Heimbigner, D., Osterweil, L. (1995). APPL/A: A language for software process programming. *ACM Transactions on Software Engineering Methodology*, Vol. 4, nr. 3, 221–286.