SOFTWARE PROCESS IMPROVEMENT AND PRACTICE
Softw. Process Improve. Pract. 2005; **10**: 379–392
Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/spip.241

Capturing Design Rationale with Functional Decomposition of Roles in Business Processes Modeling

Research Section

Pavel Balabko,[1,*,†] Alain Wegmann,[1] Alain Ruppen and Nicolas Clément
[1] *Systemic Modeling Laboratory (LAMS), School of Computer and Communication Sciences, Ecole Polytechnique Fédérale de Lausanne, Lausanne, Switzerland*

Enterprises need to create and maintain the fit between their business processes and their business process support (BPS) systems. This frequently requires re-engineering of business processes. If the business processes' design rationales are made explicit, the re-engineering of the existing business processes is easier. Making the design rationales explicit is useful to understand why things exist as they are, and to understand if and how they can be changed.

The most common way to capture design rationales is to list the design decisions underlying the business process. In the method proposed in this publication, we capture design rationales through role modeling. We document the design rationales for each role in the analyzed business process. The roles are described as the compositions of 'base functionalities'. A base functionality is the specification of a role (or of a part of a role) from the viewpoint of one of the specific specialists who designs the business process. These base functionalities are composed together to describe the roles of the participants to the business process. As a result, a business process modeler can understand who is responsible for what in the business processes. In collaboration with the specialists, the business modeler can then ensure that no important part of the business process is missing or is unnecessary. Owing to this work, the business process can be verified before being automated with the BPS system. Copyright © 2005 John Wiley & Sons, Ltd.

KEY WORDS: design rationale; business process modeling; business process re-engineering; manufacturing; role modeling

1. INTRODUCTION

In many companies, the core business process does not change much over the years. The same process can be used for several years while being slowly adapted to new technologies. During that time, people involved in the design of the business process may change. The result of these changes

* Correspondence to: Pavel Balabko, Systemic Modeling Laboratory (LAMS), School of Computer and Communication Sciences, Ecole Polytechnique Fédérale de Lausanne, Lausanne, Switzerland
† E-mail: pavel.balabko@epfl.ch

is the possible loss of the design rationales that explain the structure of the business process. This happens because the documentation of the design rationales is often considered a routine, is not made explicit and, therefore, is lost with changing business process stakeholders.

Even if the core business process does not change much, its implementation might change. A typical case is the introduction of a Business Process Support (BPS) system in order to automate a business process. To understand how an existing business process can be automated with the BPS system, it is necessary to distinguish between parts of the business process that can be automated and parts that cannot be automated with the new BPS system. The business process modeler has to be sure that she has captured all-important parts of the business process and that the business process does not include unnecessary actions that are no longer necessary to achieve the goal of the organization. This requires from the modeler a detailed understanding of the business process. Complicated business processes are difficult to understand without a clear picture of the *design rationales* that justify the existing process structure. The business process modeler has to understand *why* the process is the way it is and *how* the existing process structure was selected. If the modeler asks the employees that participate in the business process why certain design decisions were made, the most common answer would be that these design decisions have proved themselves over time: the existing business process has worked successfully for many years. In many cases, the traces that explain the design rationale are lost, and finding the people who made a certain design decision is not easy.

The lack of explicit rationale in business process models results in incorrect model: some parts of the business process may be missing or misinterpreted. The solution to this problem is to make explicit how business processes are designed. 'A minimum requirement for a design knowledge management capability is that it must be able to record historical precedence, as well as statements of beliefs and rationalizations for why a current design situation is identical to a previous one' (Mayer *et al*. 1995). Documenting design rationales is a challenge. 'It is unreasonable to expect designers to introduce a separate step in the design process to document, or model, the assumptions or rationale upon which a given design process is based. Therefore, much of the capture of this information must occur through background processes or interactive questioning initiated by a design support environment, rather than the designer' (Mayer *et al*. 1992). We propose a method in which it is easier to capture design rationales because the integration of the viewpoints of the different business process stakeholders is made explicit and is kept explicit. In the suggested method, we encourage the modeler to think about the base viewpoints (we call them functions) involved in a business process and to define explicitly how these functions are composed together into the specification of the business process. This improves the understandability of business process models, increases the reusability of model elements and facilitates business process re-engineering. The method that we introduce in this article is part of the Systemic Enterprise Architecture Methodology (SEAM). SEAM is an Enterprise Architecture (EA) method that supports the hierarchical analysis and design of enterprise systems (Wegmann 2003).

The main contribution of this article is practical rather than theoretical. We believe that the suggested method improves the understandability of business process models with the explicit modeling of the design rationale for business processes. Thinking in terms of roles/functions helps a modeler to understand the crafts of the business process stakeholders and to see how these crafts are used in the business process. To illustrate the practical outcome of the suggested method, we use the example of a manufacturing process in Pharma Co., a pharmaceutical company. This company needed to introduce a kind of BPS system called MES (Manufacturing Execution System). The project goal was to ensure traceability while reducing operation costs. In this project, the suggested method was used to ensure the correct understanding of the manufacturing process by the business process modelers and to ensure that the model of the manufacturing process correctly reflects the viewpoints of all specialists involved in this process. This was a prerequisite before deciding what information needed to be tracked by the MES.

In Section 2, we show how our work differs from others. In Section 3, we explain the proposed method with the example of Pharma's manufacturing and planning processes. We present two

applications of the method: the design of a manufacturing process and the design of a manufacturing plan. The first application illustrates in details the function composition. The second application illustrates the generation of alternatives as different function compositions. Section 4 is the conclusion.

2. OVERVIEW OF THE METHOD FOR CAPTURING DESIGN RATIONALE

Capturing design rationale has been a research topic for a long time, beginning from the early 1980s. A good introduction to this topic is in 'The Encyclopedia of Computer Science and Technology' (Buckingham Shum 1996). We define design rationale as 'The beliefs and facts, as well as their organization, that the human uses to make (or justify) design commitments and to propagate those commitments' (Mayer *et al*. 1995). The most common way to represent design rationale is to provide traceability (usually in a semiformal way) from project artifacts to design rationale arguments. According to Mayer *et al*. 1995, the design rationale arguments can be as follows:

- The philosophy of a design that describes, at a high level of abstraction, the behavior and structure of the system of interest.
- The design limitations expressed as range restrictions on systems parameters, and on environment factors.
- The factors considered in trade-off decisions, including budget constraints, timing constraints, organization policies/procedures and availability of technology.
- The design goals (including components that have to be used, priorities on problems requirements, . . .).
- The precedence of historical proof of viability.
- The legislative, social, professional, business or personal evaluation factors or constraints.

The design rationale is usually captured in the form of the design history (see the following examples: Design Space Analysis (DSA) with the Design QOC notation, proposed by MacLean *et al*. (MacLean *et al*. 1991); the Issue Based Information System (IBIS) notation and tool (Conklin and Begeman 1989; Conklin and Burgess Yakemovic 1991); Decision Representation Language (DRL) (Lee and Lai 1991) with the corresponding tool called SIBYL (Lee 1990)). The design history is represented as the network of intermediate artifacts that have to be reached in order to get the model that satisfies the overall design goal. The design history is composed of iterations. In each iteration, the designers document each artifact with the issues that have to be solved or the goals that have to be reached. Then they consider multiple alternatives to solve the identified issues or to reach the goals. Each alternative is motivated with its design rationales. The designers then choose one of the alternatives based on these design rationales. Once the adequate alternative is selected, the process is repeated until the overall design goal is reached.

In the context of business process modeling, a popular method to capture design rationale is the Design Rationale Capture Method (or IDEF6) (Mayer *et al*. 1995). This method makes explicit the traces from IDEF model elements to the aforementioned design rationale arguments. The main emphasis in this method is given to the definition of different types of the design rationale arguments (such as the ones listed above). In IDEF, the design process consists of several refinement steps, where each step is done in an *ad hoc* manner. Therefore, it may be quite difficult to understand the design rationale behind the refinement of IDEF models. In addition, the design rationale for each refinement step may include the arguments of very different business process stakeholders without making explicit who is the stakeholder. This also complicates understanding.

Our goal is to develop a design method that keeps separate and explicit the design rationale of the different business process stakeholders during the design process. For this reason, we propose a method that considers the business processes and the design processes:

- The Business Process (BP) is the process that needs to be designed or re-engineered. In the proposed method, we use the role-modeling technique for the specification of business processes: we model them in terms of the roles of the process participants. Roles represent abstractions of the behaviors of the participants related to the collaborations in which they participate. For example, a manufacturing operator plays a role in the manufacture product X collaboration.
- The Design Process (DP) is the process that results in the specification of the aforementioned

business process. In the proposed method, we model the DP as the collaboration of engineers. They define functions (artifacts) that are used for the specifications of the roles of the business process participants. For example, the role of a manufacturing operator is a composite function. This function is specified as the composition of base functions (such as the QC function or the manufacturing function).

Other role modeling methods exist such as RIN – Role Interaction Networks (Singh and Rein 1992), RAD – Role Activity Diagram (Ould 1995) or OORAM – the Object-oriented Role Analysis Method (Reenskaug 1996). These three techniques are quite similar. Roles are considered as sets of sequentially ordered actions. The method, proposed in this article, differs from the aforementioned role modeling techniques by the following characteristics:

- It takes into account not only roles from the business process, as other role-modeling techniques, but also roles from the design process. Considering the design process makes explicit crafts of the specialists (or engineers) who design the business process.
- It defines base functions and composite functions. Base functions define the base unit of composition. Composite functions represent the compositions of base functions, of composite functions or of both. The proposed method defines the notion of composition constraints[‡] They are used to make explicit how composite functions are created.
- It makes explicit the attributes (or objects) in the specification of functions and roles (Wegmann 2003; Balabko and Wegmann 2003b). The other role-modeling techniques represent roles as pure behaviors (i.e. sequences of actions, keeping the attributes implicit).

In our modeling approach, we represent processes as the collaboration (a dashed oval) of objects (cubes connected to the collaboration), see Figure 1. UML-like stickman can also represent objects (such as in Figure 2). Objects can represent companies, humans, facilities, tools or materials. Each object plays a role in the collaboration. The specification of a role is a function (visible as rectangles with squares and ovals in Figure 1). A function can be a base function or a composite function (i.e. the composition of several functions).

Figure 2, for example, represents the engineers who design a manufacturing process. Each engineer, by participating to the collaboration 'define base function', specifies the base function that he or she is responsible of as follows:

- The *manufacturing process engineer* specifies the *product manufacturing function* based on the formulas provided by pharmaceutical chemists and the available manufacturing tools.
- The *logistic engineer* specifies the *raw material provider* and *product sender functions*. These functions describe the distribution and storage of raw materials and manufactured products.
- The *manufacturing tools engineer* specifies the *manufacturing floor* function. This function describes the manufacturing tools to be used in the manufacturing process.
- The *QA* (*quality assurance*) *engineer* specifies that the *QC function*: this function describes the quality control actions in the manufacturing process.

The method proposed in this article defines a DP that has three main actions:

- '*Define the base functions*': Definition of the base functions. The design rationales for each base function are captured in a traditional way and we do not address this aspect in this article.
- '*Compose the base functions*': Definition of the alternative composite functions. Each alternative is specified with *composition constraints* that capture design rationales. Composition constraints specify the relation of dependence between functions that have to be composed.
- '*Choose the appropriate composition alternative*': Selection of the adequate alternative on which the design will be based.

Figure 3 illustrates two occurrences of this DP. The first occurrence represents the design of the manufacturing process (Section 3.1). The second occurrence represents the design of the manufacturing plan (Section 3.2). We present both processes in the same figure to emphasize that they have a

[‡] Composition constraints between composed functions make the method proposed in this article similar to Aspect-Oriented Programming (AOP) (Kiczales 1997) and Subject-Oriented Programming (SOP) (Harrison and Ossher 1993). With these approaches, objects are specified as a set of aspects/subjects that are composed together with the composition specification.

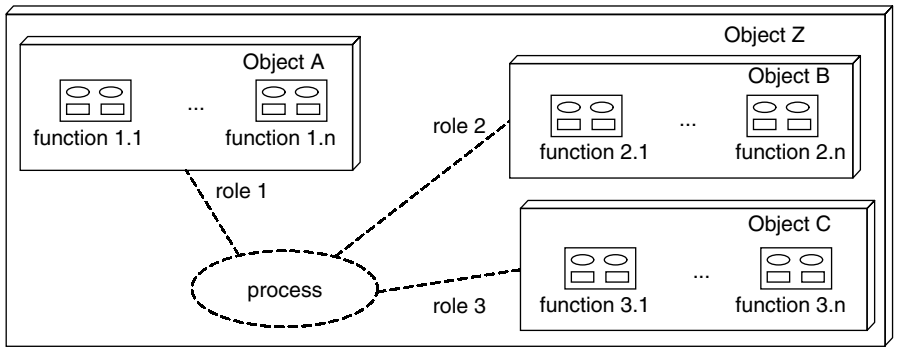

Figure 1. Business process representation in our approach

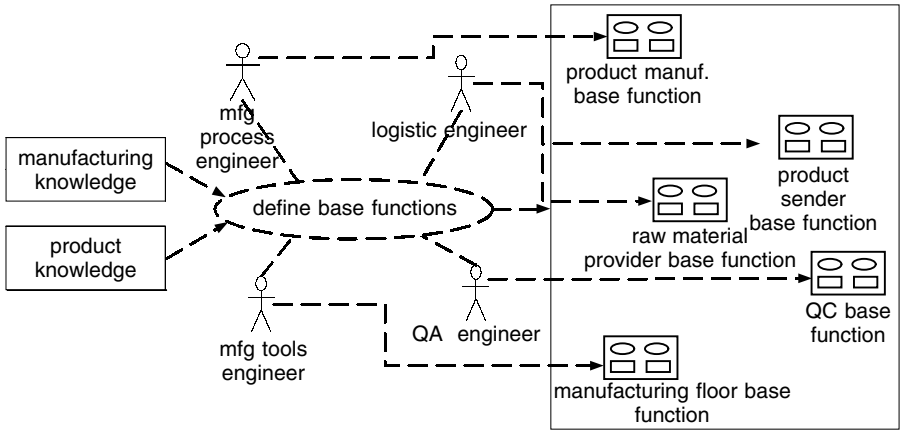

Figure 2. The 'define base functions' collaboration and the base functions defined in this collaboration

similar structure to illustrate what are there specific deliverables and to illustrate how both occurrences of the DP are related to each other.

In summary, the proposed approach contributes to the capture of design rationales by its possibility to explicitly reason in terms of composition of functions. Through this mechanism, it is possible to make explicit who is responsible of what (function) and how these functions are composed together into the resulting business process model. Our method can be applied either to engineer new processes or to analyze existing processes.

3. PHARMA CO. MANUFACTURING PROCESS MODELING WITH ROLES

In this section, we explain our method with the example of Pharma's manufacturing process and planning process. We present two occurrences of the DP presented in Section 2. In Section 3.1, we show, in detail, how a manufacturing process can be described as a composite function. We compare our approach to an IDEF-based specification. In Section 3.2, we show how the planning process could be considered as another application of the DP presented in Section 2. We illustrate how alternatives could be generated.

3.1. Manufacturing Operator Composite Function

In this section, we focus on the manufacturing operator composite function. First, we show how engineers from Pharma Co. specify this function (Figure 4). They use an internal notation inspired by IDEF (NIST 1993) and UML Activity Diagrams (OMG 2003). It consists of blocks representing the actions present in the manufacturing process.

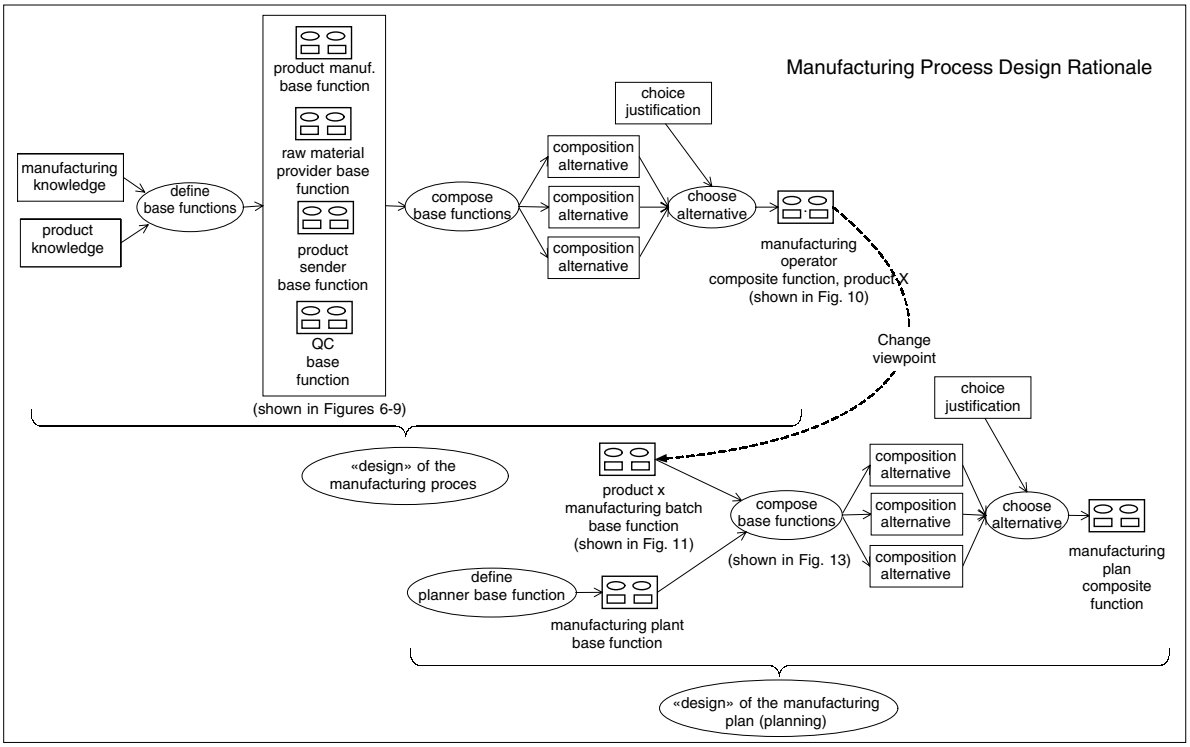

Figure 3. Overview of the DP, which makes explicit the design rationales as the composition of functions

Each block contains the following information:

- Action name and description (in the center of the block).
- Identifier of the action (upper-left corner).
- Identifier of the manufacturing room in which the action is executed (lower-left corner).
- Identifiers of the relevant subprocesses: these identifiers link each block to more detailed subprocesses specified in a similar way (lower-right corner).

Blocks are related with arrows. They represent the following:

- Horizontal incoming arrow: consumed products (names and quantities of products are given above arrows).
- Horizontal outgoing arrow: created products (names and quantities of products are given above arrows).
- Vertical incoming and outgoing arrow: sequential constraints between actions that specify the sequence of actions. Note that actions can be executed in parallel, for example, the 'Solution preparation' and 'Alcoholic solution preparation' actions are executed in parallel after the 'Raw material quality control' action.

On the basis of this notation, we can see that the operator of the manufacturing process has to occupy a manufacturing room, receive raw materials, complete the quality control of these materials and then manufacture a final product. Then, the operator has to complete the quality control by sampling the product, prepare and send the bulk and release the manufacturing room. In the following section, we illustrate how this manufacturing process can be specified as the composition of base functions, where each base function represents a view of a specific engineer of Pharma Co.

3.1.1. Base Functions in the Manufacturing Operator Composite Function

The manufacturing process represented in Figure 4 can be broken down into four base functions: raw material provider, product manufacturing, bulk preparation and quality control (QC). As described in Figure 5, these four functions are performed by

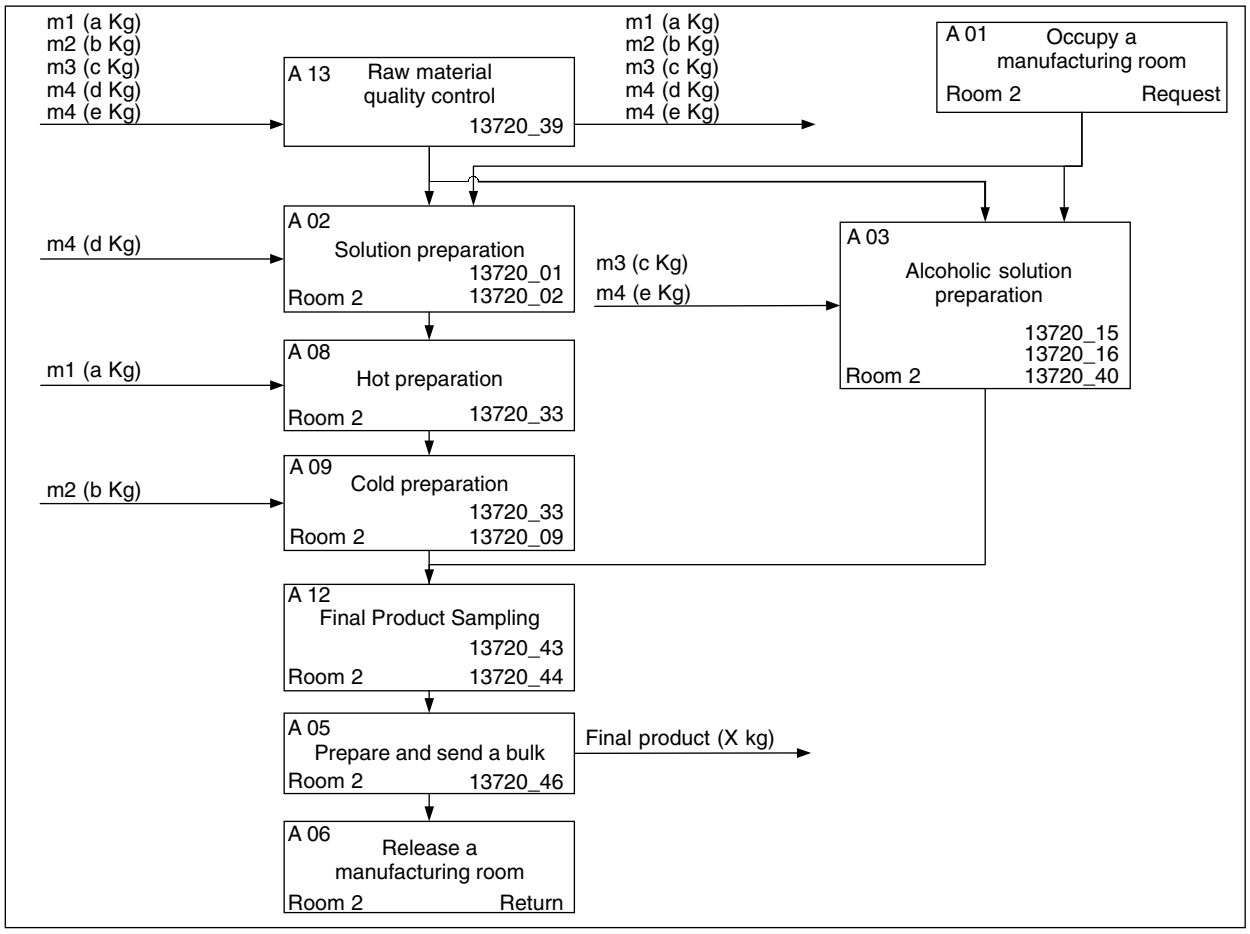

Figure 4. The manufacturing operator responsibility in pharma Co

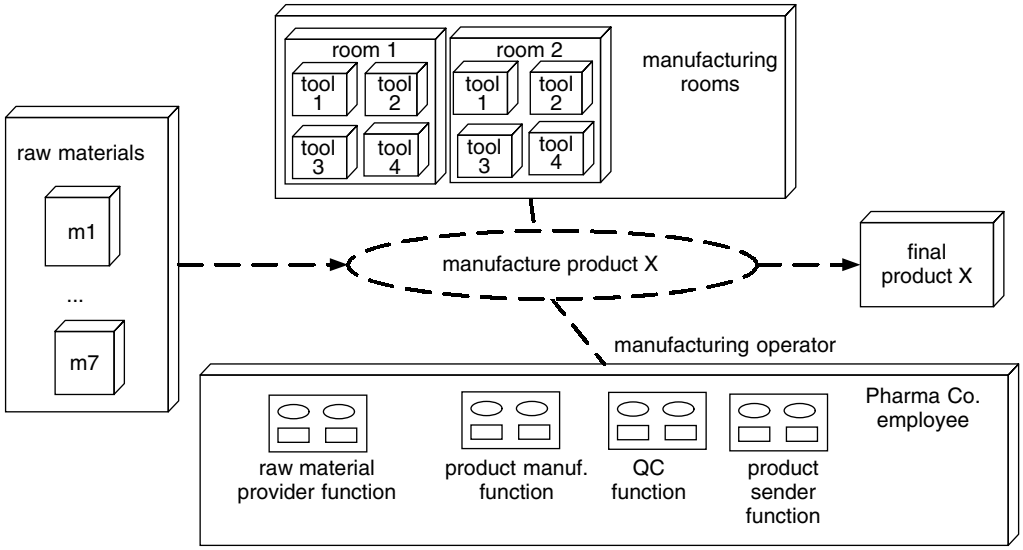

Figure 5. 'Manufacture product X' collaboration

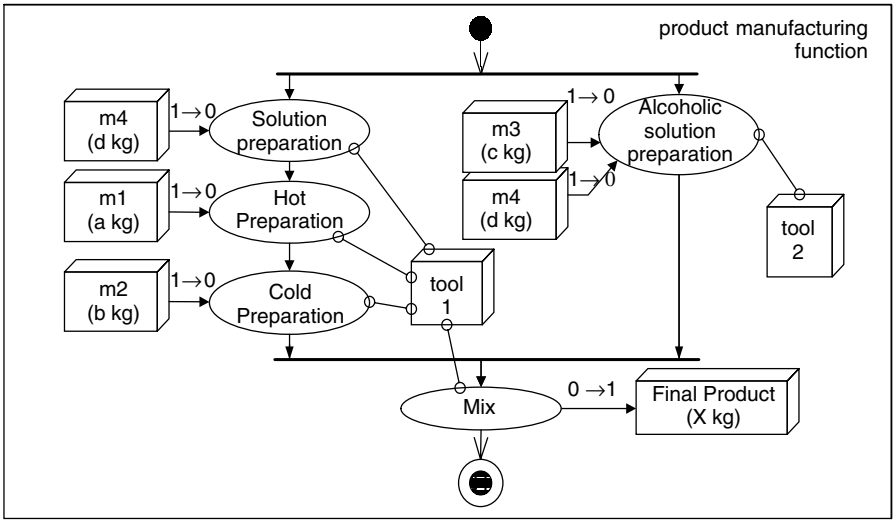

Figure 6. 'Product manufacturing' base function

an employee in the role of a manufacturing operator; this employee is responsible for manufacturing a final product from raw materials using the tools from the manufacturing rooms.

Our model of the manufacturing process makes explicit the views (base functions) of different engineers and the way these views are composed into one composite function. We now present the base functions.

3.1.1.1. 'Product Manufacturing' Base Function This function is the core function of the manufacturing process. It specifies how a Final Product is created from raw materials (see Figure 6). In this function, four actions consume raw materials. We indicate this 'consumption' by changing the multiplicity of the raw materials (mX objects) from one to zero ($1 \rightarrow 0$) and with arrows that go from the consumed materials to the corresponding actions. The 'Mix' action results in the creation of the Final Product. We indicate this by changing the multiplicity of the Final Product from zero to one ($0 \rightarrow 1$) and with the arrow that goes from this action to the Final Product. The model from Figure 6 also specifies the tools that are used in the manufacturing process. For example, Tool 2 is used in the 'Alcoholic solution preparation'. Tool 1 is used in all other actions.

3.1.1.2. 'Process Interface' Base Functions The *raw material provider* function defines the action of receiving raw materials from the warehouse. The result of this action is that multiple raw materials become available. We indicate this in Figure 7a by changing the multiplicity of the materials (m*X*) from zero to multiple ($0 \rightarrow *$) and with the arrow that goes from the 'mX' object to the 'Get raw materials' action. The *product sender* function defines the 'Prepare and send a final product' action. In Figure 7b, we indicate that the final product was send by changing its multiplicity ($1 \rightarrow 0$).

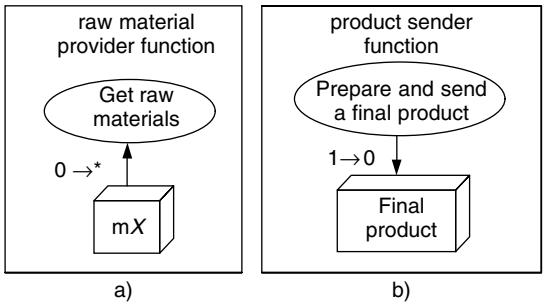

Figure 7. Base functions: (a) 'raw material provider' and (b) 'product sender'

3.1.1.3. 'QC' Base Function The QC function (Figure 8) specifies two quality control actions. The first is used for the control of raw materials before the manufacturing process (raw materials change state from unchecked to checked). The second is used for the control of the manufactured product after the manufacturing process (sampling the manufactured product).

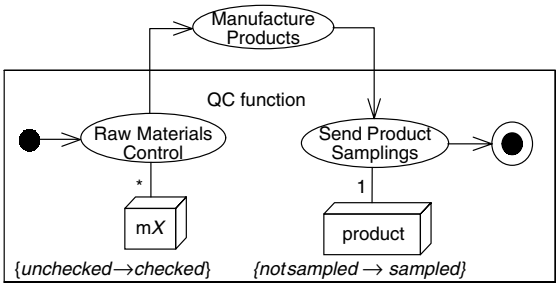

Figure 8. 'QC' base function

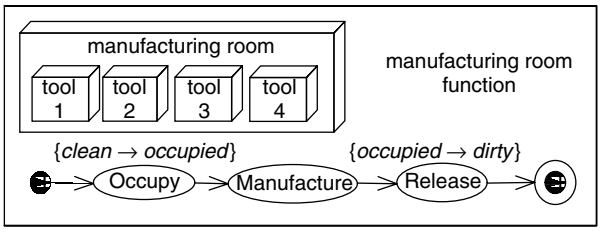

Figure 9. 'Manufacturing room' base function

3.1.1.4. 'Manufacturing Room' Base Function The main idea of the manufacturing room function (see Figure 9) is to specify the states of the manufacturing floor rooms. The manufacturing room changes its state from 'clean' to 'occupied' in the Occupy action and from 'occupied' to 'dirty' in the Release action.

3.1.2. The Manufacturing Operator Composite Function as the Composition of Base Functions

In this section, we explain how the above base functions are composed together. Figure 10 represents the result of this composition; it defines the composite function that specifies the role of the operator in the manufacturing process. The composition is made with *composition constraints:* constraints implied on the behavior of functions to be composed. Composition constraints show how the composed functions are related to each other. There are different types of composition constraints (Balabko and Wegmann 2003a). In our work we use only two types:

- *Constraints of sequentiality*: These define the sequence of actions of functions to be composed. For example, in Figure 10, the 'Get raw materials' action (Raw material provider role) precedes the 'Raw materials control' action (QC role).
- *Identity constraints*: Two model elements are identical if they represent the same entity in the reality perceived by modelers (see Balabko and Wegmann 2003a). For example, raw materials received in the 'Get raw material' action are identical with materials to be checked by the 'Raw material control' action.

In Figure 10, we show all constraints of sequentiality (we represent them with dotted arrows between constrained actions) and only some identity constraints related mainly to the QC function (we represent them with dashed lines between identical objects).

In Figure 10, it is visible that the raw material provider function generates materials that are then checked in the QC function. Once checked and when a manufacturing room can be occupied, the product manufacturing function can be performed. The resulting product is checked by the QC function and the product sender function puts it in inventory. The room can be released and is in the state dirty. Figure 10 is equivalent to Figure 4, but makes explicit why actions are performed.

3.2. Manufacturing Plan Composite Function

The example of the manufacturing operator shows how a function can be described as the composition of base functions. The model shown, even if real, is quite simple. The composition of base functions is quite straightforward: only one composition alternative is possible. Our goal in this section is to illustrate the presented DP when multiple alternatives of composition are possible. We analyze how a manufacturing plan (composite function) can be generated.

The manufacturing plan composite function defines the role of the IT system (such as an MES – manufacturing execution system) or of the employee in charge of the control of the execution of the manufacturing plan. The plan describes which manufacturing room and which operator will have to produce which set of products during a given week. The composite function can be broken down into two kinds of base functions: 'product X manufacturing batch' and 'manufacturing plant' base functions. These base functions need to be composed to define the actual manufacturing plan that needs to be executed. The corresponding DP is document in the lower part of Figure 3.

Section 3.2.1 defines the base functions necessary for the planner and Section 3.2.2 presents how these

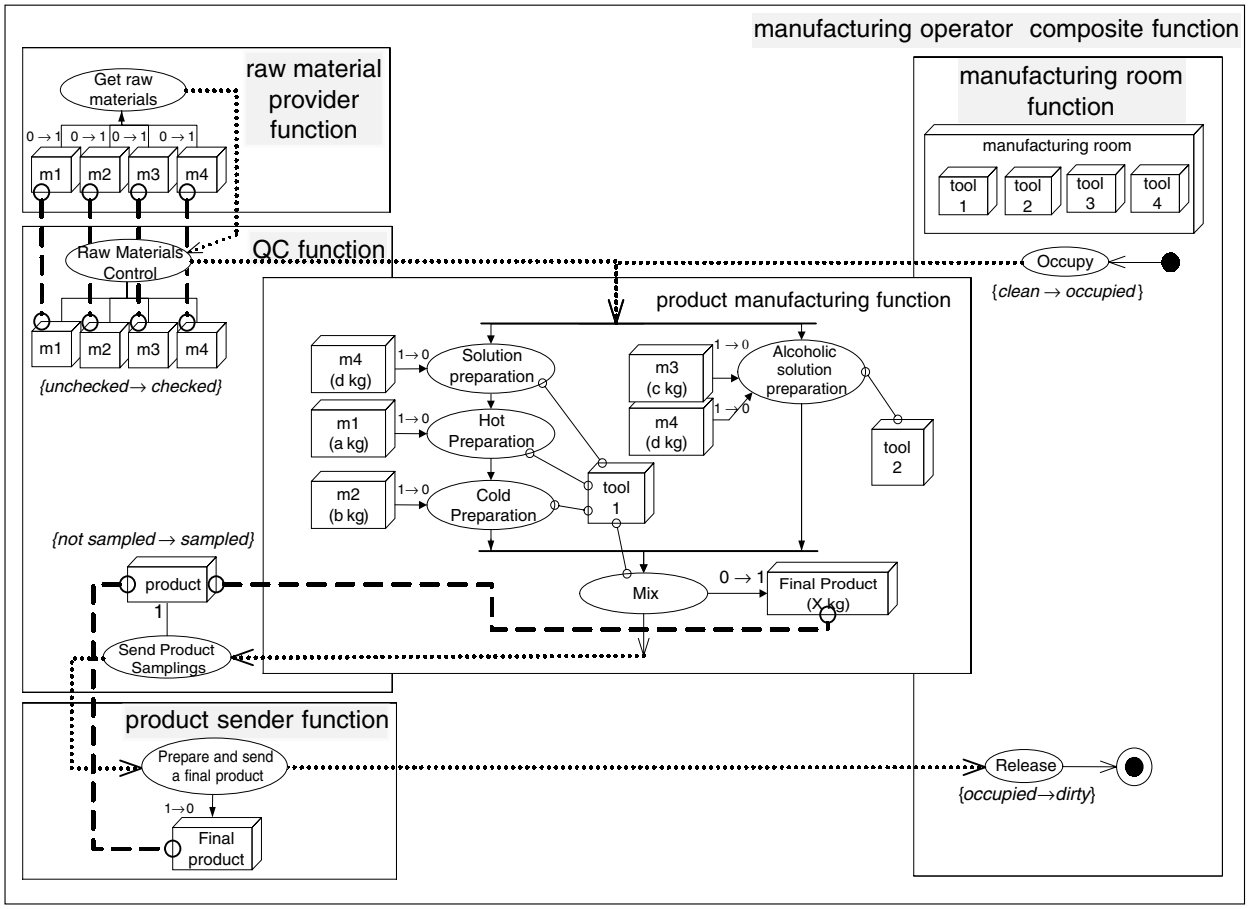

Figure 10. The 'manufacturing operator' composite function for a product X

base functions can be combined to generate the manufacturing plan. This section assumes that an MES system is used to control the production. So we use the term MES and do not reference the fact that an employee could have the same responsibility. Figure 11 illustrates the collaboration 'manufacture in week N', which describes the goal to be achieved by the planner.

3.2.1. Base Functions in the Manufacturing Plan Composite Function

The design of the manufacturing plan takes into consideration two base functions. One base function, named 'manufacturing plant base function' defines the plant equipment (in terms of rooms and employee capable to play roles). The 'Product X Manufacturing batch' is the representation, from the MES standpoint, of the 'Manufacture Product X' collaboration shown in Figure 5. The role of the planner is to compose multiple occurrences of the second function (that describes the manufacturing of a product) with the function that describes the plant capabilities.

3.2.1.1. 'Product X Manufacturing Batch' Base Function This function corresponds to the abstract view of the MES system on the 'Manufacturing Product X' collaboration as shown in Figure 5. It specifies that the product X should be manufactured by an operator Z (an available operator in a given interval of time) and in a room Y (an available room in a given interval of time) (Figure 12).

3.2.1.2. 'Manufacturing Plant' Base Function The manufacturing plant base function (Figure 13) specifies the set of manufacturing constraints that have to be satisfied in the following manufacturing plan:

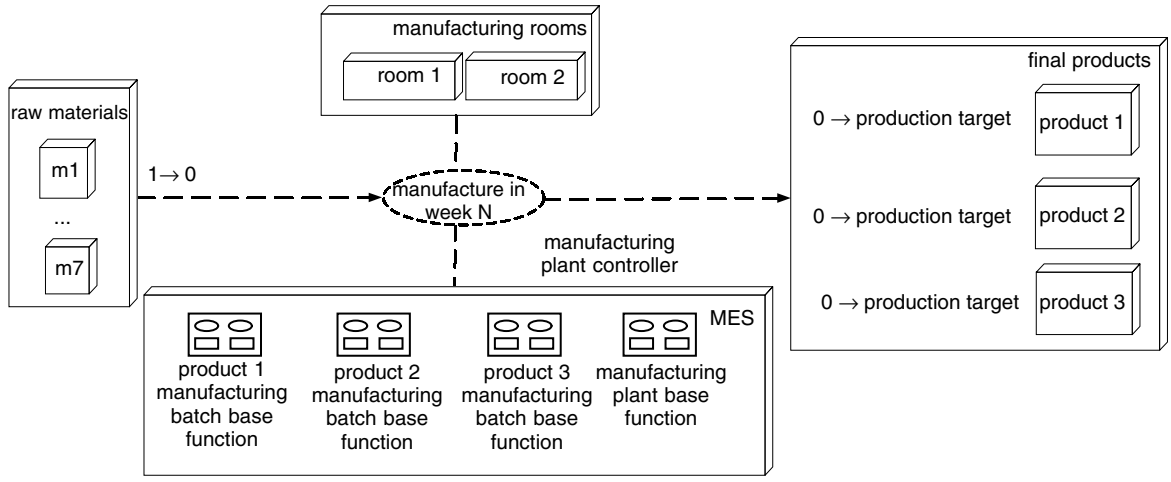

Figure 11. The 'manufacture in week N' collaboration

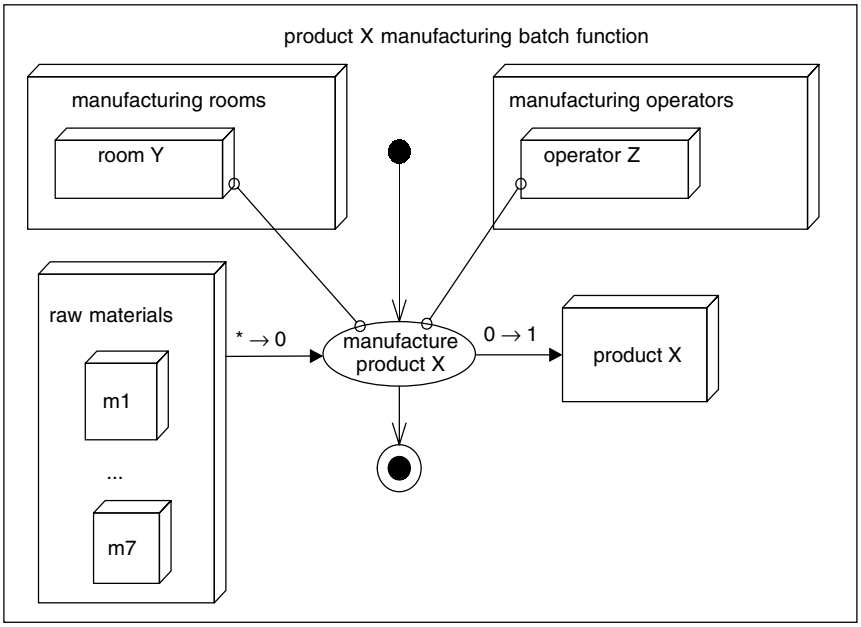

Figure 12. 'Product X manufacturing batch' base function

- Number of available manufacturing rooms on a manufacturing floor.
- Number of available manufacturing operators. The number of operators can be lower than the number of rooms. In that case, one operator has to operate several rooms simultaneously.
- Maintenance constraints: they define when the tools in manufacturing rooms have to be cleaned and maintained (to make our example easier, we do not show maintenance constraints).

3.2.2. The Manufacturing Plan Composite Function as the Composition of Base Functions

To specify the manufacturing plan composite function for the quantity of products defined in Figure 11, the production planner has to compose multiple instances of the 'Product X manufacturing batch' base functions with the manufacturing plant base function. The responsibility of the planner is to define the right sequence of these functions. Figure 14 shows the first two days of the manufacturing plan composite function. It is specified as

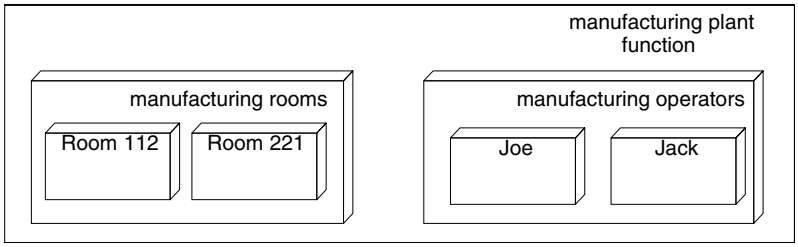

Figure 13. 'Manufacturing plant' base function

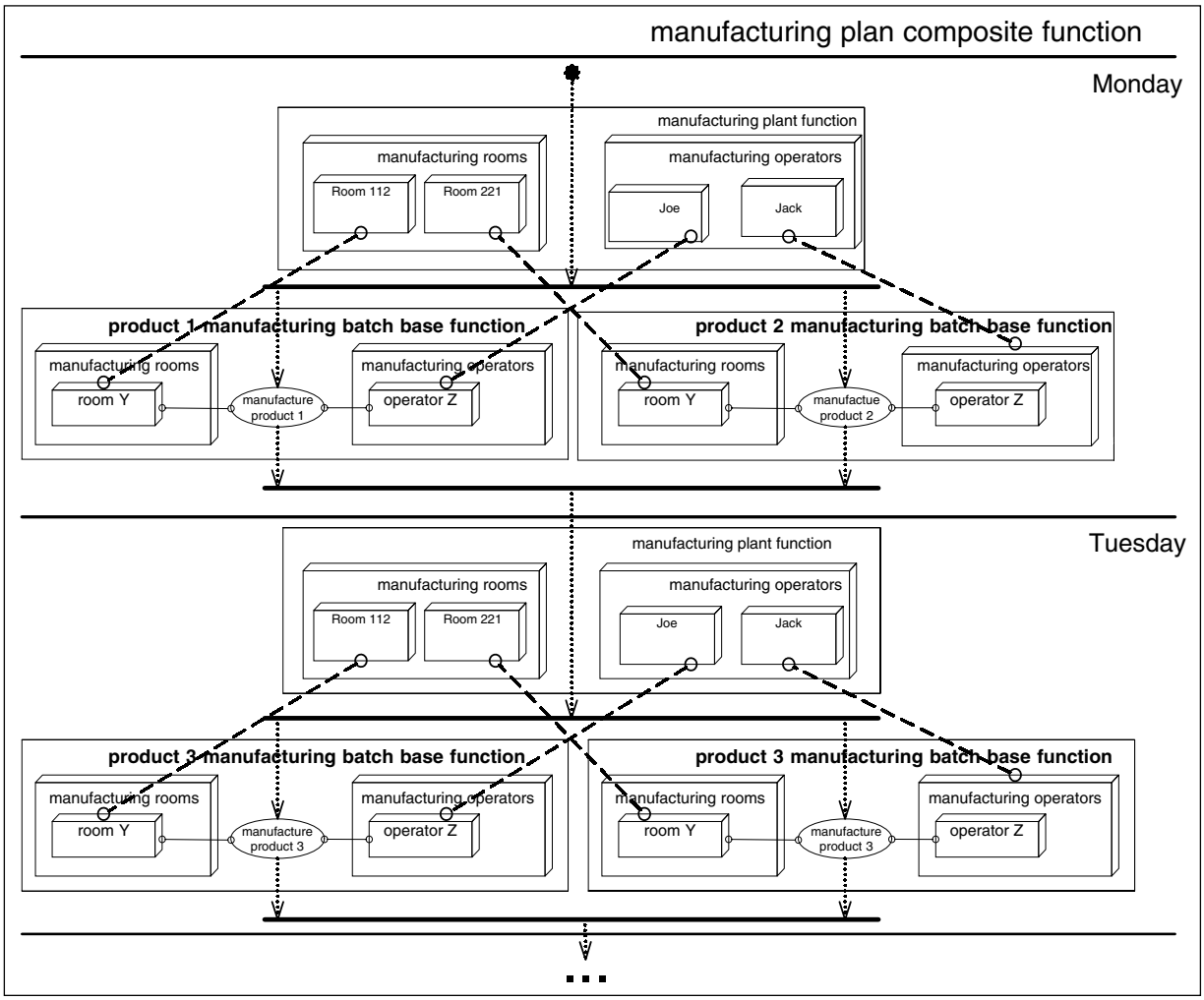

Figure 14. 'The manufacturing plan' composite function

the composition of base functions in the same way as in Section 3.1.2 (using constrains of sequentiality and identity constraints). For simplicity, we do not show the raw material and the product X objects in this diagram.

Figure 14 shows one of the possible alternatives for the manufacturing plan composite function. Other alternatives would manufacture products in a different order or change the rooms/operators involved. The planner will select the alternative

that best fits her goal (e.g. minimization of the manufacturing time, maximization of the resource usage, . . .).

4. CONCLUSIONS

In this article, we present a method that explicitly models the way business processes are designed. Capturing the explicit design rationales allows for a better analysis of the existing business process models when they need to be re-engineered to maintain or improve the fit between the business environment and the business processes. For example, in the pharmaceutical industry, the QC function is very sensitive to regulatory and technological changes. If the FDA (Food and Drug Administration) changes its regulation and requires some additional verification to be made during the manufacturing process, the process will need to be changed to maintain the fit with the new FDA regulation. Similarly, if a technological change enables the company to perform an automatic quality control where the control is manual today, the process has to reflect this change. To be able to isolate the actions directly related to the QC function and to understand how these actions are related to the ones in the other functions is crucial when the QA engineer needs to re-engineer the business process.

Our method defines base functions that represent the craft of the business process stakeholders (engineers). Every function can be separately analyzed by the relevant stakeholder. These base functions are composed into composite functions and, eventually, they specify the roles of the participants to the business processes. Our method defines composition constraints that make explicit how the functions are composed.

In our method, we identify base functions as crafts of manufacturing process specialists. However, sometimes, actions in a business process are not evidently linked to the responsibility of the identified specialists. It might also happen that some actions are missing (for example, if a company does not know the need to have a QC function). Therefore, the question is how to insure the completeness of a business process model in such cases? This can be solved in two ways:

- It is possible to define libraries of base functions. In our article, we have considered some base function (such raw material provider, quality, control product manufacturing roles) and composite functions (such as a manufacturing operator). We could develop a library of all necessary functions in a similar manner as in Bernstein *et al*. 1999.
- It is possible to identify base functions by comparing complete business processes in different companies, markets, etc. For example, it is possible to identify the base roles in the music industry by comparing the on-line music distribution to the traditional music distribution channels. (Balabko and Wegmann 2003a). This approach has an additional benefit as it is possible to think about alternative business processes for a company and this helps creativity.

The future work consists in developing a CAD tool that supports our method. Existing business process modeling tools, based on role modeling, can be used for role modeling. Examples of such tools are tool (RADRunner 2005) based on Role Activity Diagrams (Ould 1995), Graphical Re-Engineering Analysis and Design Environment (GRADE) tool (Kalnins 1996), the business process management toolkit ADONIS® 2005 and others. However, these tools do not support the composition of views (or functions). In these tools, a designer cannot apply the concept of composition constraints used in our method. Our tool will be an evolution of SeamCAD (Le and Wegmann 2004), the tool developed in our laboratory. SeamCAD is designed for hierarchical-system design. It uses a UML-like notation (OMG 2003). An additional feature is its integration with model checking tools. With this feature, it will be possible to check the base and composite functions either dynamically (by simulating the behavior) or statically (by looking at prototypical instances diagrams (Balabko and Wegmann 2003c) generated using the Alloy Model Checker tool (Jackson 2002)).

REFERENCES

ADONIS® – The Business Process Management Tool. 2005. *BOC Information Technologies Consulting GmbH*, downloaded from http://boc-eu.com/html/adonis.html on May 10, 2005.

Balabko P, Wegmann A. 2003a. A synthesis of business role models. *Proceedings of ICEIS Conference*, Angers, France.

Balabko P, Wegmann A. 2003b. Context based reasoning in business process models. *Proceedings of IEEE IRI'03*, Las Vegas, USA.

Balabko P, Wegmann A. 2003c. Precise graphical representation of roles in requirements engineering. *Proceedings of EMSISE03 Workshop*, Geneva, Switzerland, September 5, 2003.

Bernstein A, Klein M, Malone TW. 1999. The process recombinator: a tool for generating new business process ideas. *Proceedings of International Conference on Information Systems*, Charlotte, NC.

Buckingham Shum S. 1996. *The Encyclopedia of Computer Science and Technology*, Vol. 35, Suppl. 20. Marcel Dekker: New York, 95–128.

Conklin J, Begeman ML. 1989. gIBIS: a tool for all reasons. *Journal of the American Society for Information Science* **40**: 200–213.

Conklin J, Burgess Yakemovic KC. 1991. A process-oriented approach to design rationale. *Human-Computer Interaction* **6**(3&4): 357–391.

Harrison W, Ossher H. 1993. Subject-oriented programming (a critique of pure objects). *Proceedings of OOPSLA'93*. ACM Press: Washington D.C., USA, 411–428.

Jackson D. Software Design Group. 2002. *Micromodels of Software: Lightweight Modelling and Analysis with Alloy*. MIT Lab for Computer Science: Cambridge, MA, USA, downloaded from http://sdg.lcs.mit.edu/alloy/reference-manual.pdf.

Kalnins A, Barzdins J, Auzins A, Etmane I, Kalis A, Podnieks K, Tenteris J, Vilums E, Zarins A. 1996. Business modeling language GRAPES-BM and related CASE tools. *Proceedings of Second International Baltic Workshop ''Databases and Information Systems''*, Vol. 2: Tallinn, Estonia, 3–16.

Kiczales G, Lamping J, Mendhekar A, Maeda C, Lopes C, Loingtier JM, Irwin J. 1997. Aspect-oriented programming. *Proceedings of ECOOP'97*. Springer-Verlag: New York, NY, USA, **1241**: 220–242.

Le LS, Wegmann A. 2004. SeamCAD 1.x: user's guide. Technical Report No. IC/2004/98, Enoch Partt Free Library, Lausanne, Switzerland.

Lee J. 1990. SIBYL: a tool for managing group design rationale. *Computer Supported Cooperative Work*, Los Angeles, CA, ACM Press: New York, 79–92.

Lee J, Lai K. 1991. What's in design rationale? *Human-Computer Interaction* **6**(3 & 4): 251–280.

MacLean A, Young RM, Bellotti V, Moran T. 1991. Questions, options, and criteria: elements of design space analysis. *Human-Computer Interaction* **6**(3 & 4): 201–250.

Mayer RJ, Painter M, deWitte P. 1992. *IDEF Family of Methods for Concurrent Engineering and Business Re-engineering Applications*. Knowledge Based Systems, East College Station, TX, USA.

Mayer RJ, John WC IV, Ronald F, Arthur K, Michael K. *Painter Information Integration for Concurrent Engineering (IICE): Compendium of Methods Report*, Report, COMMAND WRIGHT-PATTERSON AIR FORCE BASE, OHIO 45433-7604, 1995.

NIST. 1993. *IDEFØ – Standard for Function Modeling in FIPS Publication 183*. NIST: retrieved from http://www.idef.com/idef0.html on March 20, 2004.

OMG. 1999. *Unified Modeling Language Specification*, Version 1.5, Object Management Group, Inc., Needham, MA, USA, downloaded from http://www.omg.org/technology/documents/formal/uml.htm accessed on August 05, 2005.

Ould MA. 1995. *Business Processes: Modeling and Analysis for Re-engineering and Improvement*. John Wiley & Sons: Beverly Hills, CA.

RADRunner. 2005. *A Better way to Support Collaboration*. Role Modelers Limited: Nunney, Somerset, UK downloaded from http://www.rolemodellers.com on May 10, 2005.

Reenskaug T. 1996. *Working With Objects: The OOram Software Engineering Method*, Wold P, Lehne OA (eds.). Manning Publications: Greenwich, CT, USA.

Singh B, Rein GL. 1992. Role interaction nets (RINs): a process description formalism. Technical Report CT-083092, Microelectronics and Computer Technology Corporation, Austin, TX, USA.

Wegmann A. 2003. The systemic enterprise architecture methodology (SEAM). *Proceedings ICEIS Conference*, Anger, France.