On the Use of a Role Ontology to Consistently Design Business Processes

Artur Caetano, José Tribolet

On the Use of a Role Ontology to Consistently Design Business Processes

Artur Caetano[1,2], José Tribolet[1,2]

[1] Department of Computer Science and Engineering, Instituto Superior Técnico, Technical University of Lisbon, Av. Rovisco Pais, 1049-001 Lisboa, Portugal
[2] INESC ID & INESC INOV, Rua Alves Redol 9, 1000-029 Lisboa, Portugal
{artur.caetano, jose.tribolet}@ist.utl.pt

Abstract. The functional decomposition of a business process breaks it down into progressively less granular activities. Decomposition contributes to the modular design of a system, the reuse of its parts and to its overall comprehensibility. But achieving these qualities requires a business process to be decomposed consistently, which implies it is always split into an identical set of activities according to a specific purpose, regardless of the modeller's and modelling context. This paper describes an application of the principle of role-based separation of concerns to consistently decompose a business process into its constituent atomic activities, thus separating its distinct features and minimizing behaviour overlap. An activity is abstracted as collaboration between role types that are played by entities. The decomposition method successively separates the overlapping roles until an activity is specified as a collaboration of an orthogonal set of role types. The method facilitates the consistent decomposition of a business process and the identification of its atomic activities. The relevance of the method is assessed through a number of scenarios according to the guidelines of design science research.

Keywords: business process modelling, functional decomposition, separation of concerns, enterprise architecture.

1 Introduction

It is widely accepted that one of the fundamental problems in the design and development of knowledge-based systems is extracting information from the experts and then translating it to the form of some knowledge base in order to attain a given purpose. As in the case of business process modelling, this transformation is not straightforward as the source knowledge is often not structured or formalized and tends to be of complex nature. Furthermore, the purpose of the model itself may not be well defined or understood by all of its stakeholders. As a matter of fact, a number of researchers posit that complexity is an essential property of design activities in general due, in part, to the inevitably incomplete formulation of the problem and to our inability to cope simultaneously with all of the constraints of a given problem. Service-oriented architecture is an architectural style for constructing systems from a set of universally interconnected and interdependent services. A service is a unit of

functionality that some entity makes available to its environment. This style of architecture promotes reuse at a macroscopic service level and can simplify the usage and interconnection the business, application and technological assets within and across organizations [1]. Service orientation promotes a layered view of an enterprise architecture's models. Service layers provide functionality to higher layers and are realized in lower implementation layers. For instance, the ArchiMate enterprise modelling language [2] defines three layers: the business layer defines which products are offered to external customers through business processes; the intermediate application layer supports the business layer with application services which are realised by application components; finally, the technology layer offers infrastructural services needed to run applications.

A business process is a set of interrelated value-adding activities [3]. Activities are often modelled as opaque transformation functions that map inputs to outputs. This abstraction strategy models an activity as a black-box and focus on its external behaviour. The resulting models conceptually divide a business system into a hierarchy of functions [4]. Thus, functionally decomposing a business process entails its recursive separation into a set of more detailed activities.

Business process models translate the knowledge about how an organization operates. These models are fundamental to enterprise architecture as they support the communication, analysis, implementation and execution of the organization's structure, business, systems and technology [2, 5]. Process models also provide the means to analyze alternative designs intended to align the business with the information systems and technology. However, the process modelling must cope with the multiple views and goals of the different organizational stakeholders. Moreover, the elicitation, modelling and analysis of the processes of an organization is often the result of merging the partial contributions of different teams, probably with different backgrounds and experience. Put together, these factors lead to models that lack consistency. Examples of inconsistency include using different levels of modelling detail and the incoherent naming of the activities and entities of a process. Inconsistent process models are not only hard for their users to understand but also hamper the tasks of process analysis and redesign as they may leave out relevant information or lead to erroneous or ambiguous interpretations of the process content. Such inconsistency also negatively contributes to the identification and the specification of the services that are required to support the process.

Consistent business process decomposition can significantly improve the clarity and the overall model integrity as well as minimizing the omission of relevant information [6]. Decomposition is also a means to modularize large systems and to facilitate the reuse of partial models and favours the compactness of a specification as it allows multiple levels of detail to co-exist and coupling to be reduced [7]. As a consequence of abstraction, models become easier to understand and communicate, which, in turn, make their validation, redesign and optimization easier.

This paper proposes using the separation of concerns principle to facilitate the consistent decomposition of a business process and the unambiguous identification of its atomic activities thus contributing to the task of identifying the supporting services. To do so, we present a method that specifies how to decompose a business process according to the concerns that are involved in the specification of its activities.

The remainder of this paper is structured as follows. The next section reviews related work. Section 3 introduces the concepts of natural type, role type and activity. Sections 4 and 5 describe the functional decomposition method and the underlying role ontology along with a running example. Finally, section 6 summarizes the research methodology and section 7 summarizes the proposal and provides an outlook on future work.

2 Related Work

Functional decomposition is supported at language level by most process modelling languages, including ArchiMate [2], BPMN [8], EPC [9] and IDEF-0 and IDEF-3 [10]. The decomposition of subsystems through the hierarchic classification of process models has also been applied to Petri nets [11] and BPEL [12]. Although these approaches make possible creating a hierarchical representation of a process, their intent is not the definition of techniques for consistent activity decomposition but, instead, the representation of generic decomposition structures. Nevertheless, the shortcomings of the lack of consistency in process decomposition and in the identification of its atomic activities are pointed out by several authors [13, 14].

Several top-down decomposition approaches exploit reference models to describe how a process is structured as a hierarchy of activities. For instance, the Supply-Chain Operations Reference model describes three levels of detail to assist the definition and configuration of an organization's supply chain [15]. The Process Clarification Framework defines a hierarchical (and static) decomposition of business processes which is 3 to 4 levels deep and crosses 12 operating and management categories [16]. Other approaches, such as the ARIS framework [9], describe processes as chains of events and tasks and prescribe the levels of detail for decomposition. The first two decomposition levels address the business viewpoint of the model, the next 3 to 4 levels focus on the structure of process operation and the lower level describes the procedural details of the tasks. However, the contents of these levels of detail are actually arbitrary.

An alternative avenue of research relies on algorithmic methods to analyse the process specification and assess its consistency. One of these methods uses similarity measures derived from the syntactic and structural features of the process to detect inconsistencies between its activities [17]. These measures make use of a linguistic ontology to evaluate the similarity between the names of the activities thus assisting the detection of decomposition anomalies. Process mining techniques extract information from existing event logs and enable the discovery of business processes [18]. These bottom-up mining techniques support the verification of the conformance of a model derived from an event log against an existing model as well as identifying the atomic activities of a process [19]. Other approaches that use ontologies to specify business processes (e.g. [20-22]) also lack the means to identify atomic activities and to consistently decompose a process.

Altogether, and to the best of our knowledge, existing approaches do not define the necessary means to consistently decompose a business process and to unambiguously

identify the atomic activities that constitute it. The primary goal of this paper is therefore to provide a contribution to this research subject.

3 Fundamental Concepts

Role modelling is a separation of concerns technique that is used in multiple areas of knowledge such as data modelling [23], object-oriented and conceptual modelling [24-26], framework design [27], business process modelling [28, 29] and enterprise architecture [20, 30-34].

With role-based business process, modelling an activity (a business verb) is abstracted as a set of collaborations between entities (business nouns). The entities represent the things that are of interest within a specific modelling context. Each unit of behaviour of an entity is abstracted as a role and, as a result, activities are defined by a role collaboration pattern. If no roles are being played within the system then there are no collaborations and, therefore, no activities to be modelled.

Figure 1 shows the relationships and cardinalities between four entities involved in the `assemble product` process which we will use as a running example to illustrate the concepts outlined above. The activity `assemble product` is defined by the collaboration pattern between the roles being played by the entities `part`, `assembling machine`, `product` and `person`. The activity describes how a `product` is assembled from a number of `parts` by means of an `assembling machine`. The activity is semi-automated as the `machine` is operated by a `person`.

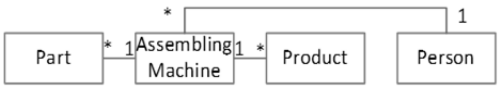

Figure 1. Relationships between entities.

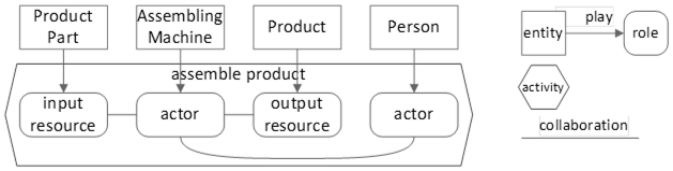

Fig. 2. Role-based specification of the assemble product activity

Fig. 2 shows the relationships between the entities result in one collaboration context where a natural type displays a specific behaviour [33, 34]. Such behaviour is abstracted as a role type. Thus, in the first collaboration context each `part` plays the role of `input resource` in their relationship with the `assembling machine` which, in its turn, is playing the `actor` role. In other context the `assembling machine` produces the assembled `product`, i.e. the `product` is the `output resource` of this `actor`. Finally, the `person` relates to the `machine` as its `actor`. The collaboration between these four roles uniquely defines the `assemble product` activity as depicted in **Fig. 2**. The `actor` role states that an entity is able to perform some action in the context of an activity. The

`resource` role states that an entity which is playing it can be used or consumed (`input resource`) or created (`output resource`) during the performance of an activity.

The remainder of this section details the concept of entity (or natural type), role (or role type) and activity.

3.1 Natural Types and Role Types

Sowa [35] distinguished between *natural types* "that relate to the essence of the entities" and *role types* "that depend on an accidental relationship to some other entity". By developing Sowa's ideas further, Guarino presented an ontological distinction between these two types [36]. This distinction is based on the concepts of *foundedness* and *semantic rigidness*. A type is considered *founded* if its specification implies a dependency or relation to some other individual. A type is *semantically rigid* if the identity of an individual depends on the type assigned to it. If the type is removed from the individual then it cannot be further identified nor classified. Thus, a type is *not semantically rigid* if it can be assigned to and removed from an individual without changing its identity. Based on the above, a type that is both *founded* and not *semantically rigid* is a *role type*. In contrast, a natural type is characterized by being *semantically rigid* and *not founded*.

To illustrate the above classification properties, let us take the example of **Fig. 2** and classify the concepts of `person` and `actor` as either natural or role types. First, let us focus on the "foundedness" of these concepts. `Actor` is a founded type since for something or someone to be assigned the `actor` type there must be something being acted upon. Conversely, the `person` type is not founded since it exists on its own terms. It defines the identity of the individual to which it is assigned to, regardless of its relationships with any other individual. Thus, the `person` type is not founded whereas the `actor` type is founded.

Regarding "semantic rigidness", the `actor` type is not semantically rigid because its identity is independent of the individual to whom the type is assigned to. This means the `actor` type is not able to identify the individual by itself. On the other hand, the `person` type is semantically rigid as its identity is directly coupled to the individual's identity. Therefore, `actor` is a role type (founded and not semantically rigid) whereas `person` is a natural type (not founded and semantically rigid).

Natural Types
Entities are natural types. In enterprise modelling, an entity describes a thing that an organization deems relevant to specify in order to fulfil the purpose of a model. Entities model concepts such as persons, places, machines, resources, contracts and products. According to the definition of natural type, an entity can be unambiguously identified and defined in isolation, i.e. without any relationship with other types. Entities can be classified according to its intrinsic features. Entities may relate structurally to other entities (e.g. an order is composed of items).

Role Types
A role type, or role for short, is the observable behaviour of an entity in the scope of a specific collaboration. Different roles separate the different concerns that arise from the collaborations between entities. Hence, a role represents the external visible features of that entity when it collaborates with another entity in the context of an activity. An entity relates to other roles through the *play* relationship. An entity that plays no roles is not participating in any activity since it is not able to produce actual behaviour. An entity enters the role when it starts playing it and leaves the role when the specific behaviour specified by the role is concluded. Each role adds a set of external features to an entity in the context of that collaboration. This effectively separates the entity's feature space since its intrinsic features are differentiated from each of the external features that transiently relate to an entity through the roles it plays.

Activities
A business process is an ordered execution of activities that produces goods or provides services that add value to the organization's environment or to the organization itself. Thus, modelling a business process involves specifying the set of activities that define its operation and the flow that defines how the activities are coordinated.

An activity is specified by a collaboration of role types. It is a behaviour element that describes part of the functionality available to the organization. Since a role type separates the description of the intrinsic features of an entity from the features that derive from the collaborations it participates in, the specification of an activity itself is independent of the specification of the entities playing the roles.

Fig. 2 depicts the `assemble product` activity as a unit of functionality that result from the collaboration between a set of roles. However, this activity model is conceptual as it may have been specified from a different perspective or with a different level of detail, which would have implied using a different role ontology. The granularity level of the activities is also arbitrary as it is always possible to add more detail to its specification. Hence, the naming of an activity is actually irrelevant for the purpose of its specification as the role collaboration pattern is the only means to specify it unambiguously. Therefore, an activity is uniquely identified by the collaboration of roles that are involved in its specification. Two activities are deemed *equivalent* if and only if they share the same set of role collaborations.

4 Functional Decomposition

The functional decomposition of a business process yields a set of sub-activities, each of which can be further decomposed. The behaviour of a whole process can then be constructed upwards from the lowest level of decomposition towards the top-level activity. The lowest level of decomposition describes primitive or atomic activities that cannot be further divided. The related literature (cf. section 2) describes different approaches to the functional decomposition of processes but, to the best of our knowledge, existing approaches do not provide the means to unambiguously identify

what makes an atomic activity nor the mechanisms that provide consistent decomposition results.

The approach proposed in this paper is to use role types as the criteria for process decomposition. This means each decomposition step separates a different concern (i.e. a role type) from the other concerns that specify the activity. An activity is deemed atomic, meaning it cannot be further decomposed, when all of its concerns are effectively separated. This translates to having no overlapping role types in the activity's specification. It also implies that the classification of an activity as atomic actually depends on the role ontology that is being utilized to generate the process model. So, different role ontologies yield different decomposition criteria and, thus, different process models.

```
decompose(S, R)
        D ← ∅
        decompose' (S, R, D, 1)
        decompose ← D
end
```

```
decompose' (S, R, D, level)
        if R ≠ ∅ then
            R0 ← firstElementOf(R)
            Dlevel ← ∅
            if numInstancesOfType(R0, S) > 1 then
                for all r ∈ R0 do
                    Sd ← (S − R0) ∪ r
                    Dlevel ← Dlevel ∪ Sd
                    decompose' (Sd, R−R0, D, level+1)
                end for
            else
                decompose' (S, R − R0, D, level+1)
            end if
            D ← D ∪ { Dlevel }
        end if
end
```

The algorithm `decompose(S, R)` recursively separates an activity into sub-activities as long as there are overlapping concerns. `S` is the ordered set of all the roles type instances used in activity to be decomposed. The set `R` (which is a subset of the types of `S`) contains the role types that define the domain to be used to decompose the activity. If all the role types in `S` are included in `R` then all roles will be separated. The role types not included in `R` will remain overlapped after the decomposition. The output of `decompose(S, R)` is a set of sets. Each of these sets represents an activity, with the outer set representing the first level of decomposition. The symbol `level` identifies the current decomposition level with 0 representing the top level activity. The symbol `D` represents the output set of the decomposition and D_{level} is the set of decomposed activities pertaining to a given `level` of depth. The algorithm makes use of two additional functions not detailed here: `firstElementOf(X)` returns the first element of the set X; `countInstancesOfType(t, X)` counts the number of instances of the type `t` within the set `X`.

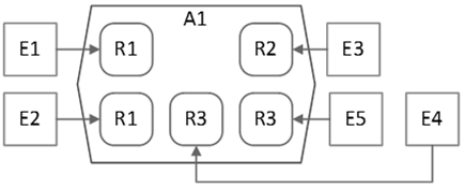

Fig. 3. Activity A1 according to roles R1, R2, R3

Fig. 3 illustrates an application of the `decompose` function to activity `A1`. `A1` is defined by the collaboration of role types `R1, R2, R3`. Let us consider that `A1` is specified by `S = {a:R1, b:R1, c:R2, d:R3, e:R3}` and that `S` maps to three role types, `R = {R1, R2, R3}`. Using `decompose(S, R)` to decompose `A1` according to `(R1, R2, R3)`, results in `D = {D1, D2}`. `D1` is the first level of decomposition and divides A1 into `{(a:R1, c:R2, d:R3, e:R3), (b:R1, c:R2, d:R3, e:R3)}`. `D2` is the lowest level of decomposition and comprises four atomic activities: `{(a:R1, c:R2, d:R3), (a:R1, c:R2, e:R3), (b:R1, c:R2, d:R3), (b:R1, c:R2, e:R3)}`.

If we define the role ontology `R1, R2, R3 to` describe locations, goals and actors, so that `R1` stands for the `Locator` role, which describes a geographical location, `R2` is the `Goal` role, that models the intended state of the affairs to be achieved after executing the activity, and that `R3` is the `Actor` role, which describes the action of someone of something operating in the context of the activity `A1`, we would get the model depicted on **Fig. 4**.

Decomposing `A1` according to the `Locator` role (`R1`) yields two activities, `A1.1` and `A1.2`, as shown in **Fig. 5**. Each of these functionally separate `A1` according to geographical location concern. Decomposing `A1` according to the `Actor` role (`R3`) produces two activities, each focusing on the specific operations of the actor involved in `A1`. Note that `A1` cannot be decomposed according to the `Goal` role (`R2`) as this concern does not overlap with any other role of the same type. Activities `A1.1` and `A1.2` can be further separated as shown in **Fig. 7** and 7.

The decomposition of `A1` according to the role tuple `(Locator, Actor, Goal)` results in four atomic activities, each focusing on a different concern: `A1.1.1 (Office:Locator, Person:Actor, Goal:Goal)`, `A1.1.2 (Factory:Locator, Person:Actor, Goal:Goal)`, `A1.2.1 (Office:Locator, Machine:Actor, Goal:Goal)`, `A1.2.2 (Factory:Locator, Machine:Actor, Goal:Goal)`. Note that `A1` cannot be further decomposed according to these three roles. Further decomposition is only possible if new roles are added to the ontology or additional overlapping concerns are included in the specification of `A1`.

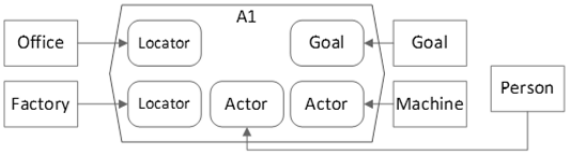

Fig. 4. Activity A1 and Actor, Locator and Goal role types

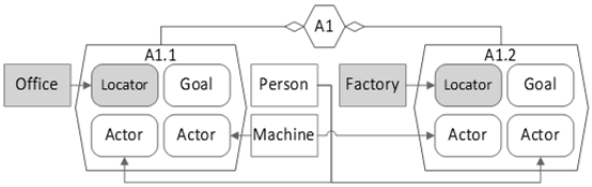

Fig. 5. Decomposition of activity A1 on role R1 (Locator)

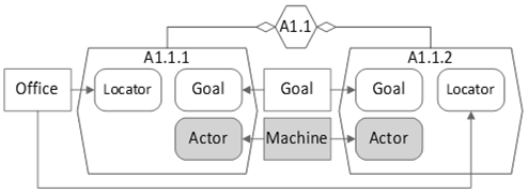

Fig. 6. Decomposition of A1.1 on role R

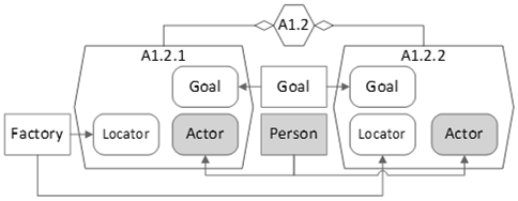

Fig. 7. Decomposition of A1.2 on role R

This approach is unambiguous as each level of decomposition can be systematically reproduced. A business process can always be consistently separated into its constituent atomic activities and the corresponding supporting services identified. Additionally, the condition for activity decomposition is explicit as the procedure stops whenever the concerns of an activity are effectively separated. Thus, consistent process decomposition promotes service identification and reuse.

5 Role Ontology

The decomposition method relies on the specification of a role type ontology. An ontology is a formal representation of a set of concepts within a domain and the relationships between those concepts. In this particular case, the ontology represents the set of role types required to model a specific domain and the possible collaborations between these role types.

A business process can be modelled from different perspectives according to the model's goals and purpose as defined by its stakeholders. Although there are multiple classification schemes to categorize the modelling perspectives, these often crosscut the six orthogonal linguistic interrogatives (how, what, where, who, when, why). These interrogatives can be used to construct four basic modelling perspectives [37,

38]. The functional perspective represents *what* activities are being performed in the context of a given process. The informational perspective represents *what* informational entities (i.e. data or resources) are being manipulated by the activities of a process. The behavioural perspective represents *when* activities are performed and *how* they are performed, usually through the specification of the process orchestration. Finally, the organizational perspective represents *why* an activity is being performed, *where* it is performed and *by whom*.

The remainder of this section exemplifies a set of roles types that addresses the above concerns according to the six interrogatives. We emphasize that the role ontology should be specified according to the requirements of the stakeholders and to the specific domain being modelled.

Actor (Who)

The `actor` role represents the action of an entity that does some task in the context of an activity. Actors are played by entities which represent people, computer systems, mechanical tools or any other devices that produce active change within an organization. A specialization scheme of the `actor` role type focuses on its nature, such as: `social actor` (people or organizations), `application actor` (computational or non-computational applications that are used to perform a task) and `infrastructure actor` (computer hardware, machines and other devices that support the application and social actors). Another specialization scheme, which is orthogonal to the actor's nature, includes roles such as `operator`, `auditor` and `supervisor`. Using the actor role as the criterion for decomposition identifies atomic that describe the actions of each individual actor. The decomposition of the `assemble product` activity in **Fig. 2** according to the `actor` role identifies two activities: one for the actions being performed by the `person` and other for the actions of the `machine`.

Resource (What)

A `resource` is the role played by an entity when manipulated by an `actor` in the context of an activity. A resource specialization scheme that focus on how a resource is transformed within an activity consists of two roles: `input resource` role and `output resource` role. The former can be further specialized as `consumed resource` role and `used resource` role, whereas the latter can be specialized as `created resource` role and `refined resource` role. Other orthogonal schemes are possible, such as classifying a resource according to its existence (e.g. tangible, intangible, etc.)

Locator (Where)

The `locator` role captures the geographical or the logical location of an entity. The sub-activities of an activity that is decomposed according to the `locator` role are operated in different locations.

Goal, Rule (Why)

A `goal` represents a measurable state of affairs that the organization intends to achieve. The entity plays the `goal specifier` role which relates to the `goal fulfiller` role. Goals are usually achieved by the entities playing the `actor` or `resource` role. A

`rule` asserts conditions or operating parameters that an activity must comply with. The entity that specifies the constraint plays the `rule specifier` role which relates to the `rule complier` role.

Starter, Finisher (How, When)
The behavioural perspective can be captured through the `starter` and `finisher` roles. The first models the event that triggers the start of an activity while the second signals its completion. These two roles can be used to describe how the activities of a process are orchestrated, as described in the next section.

6 Research Methodology

The methodology behind the results reported in this paper is grounded on design science [39, 40]. Design science focuses on the development of solutions for practical problems. This contrasts with the development and the verification of theories as in behavioural science methodologies.

Research on enterprise architecture, modelling and engineering fits the design science paradigm as its focal goal is not building information systems but creating methods and techniques to analyze, model, and understand the horizontal and vertical interfaces between the business, systems and technology [41]. The essential tangible result of a design science project consists in creating an artefact that addresses a particular issue that is relevant to a certain group of stakeholders. In this context, Hevner et al. proposed a set of guidelines to conducting design science projects [40]. The following points briefly summarize how these were applied to this work:

- **Design as an artefact.** This project deals with applying the principle of separation of concerns to business process modelling. This paper describes an artefact that deals with business process decomposition role modelling as a separation of concerns mechanism.
- **Problem relevance.** The artefact enables the consistent decomposition of a business process. By doing so, it addresses several problems that are relevant in enterprise engineering in general and business process modelling in particular. We emphasize the following problems: (1) how to systematically identify the atomic activities of a process; (2) how to make explicit the principles behind process decomposition; (3) how to make decomposition dependent on the specification of the process and not on the modelling team experience
- **Design evaluation.** This paper makes use of a *scenario* [40] built around the artefact to demonstrate its application and utility.
- **Research contributions.** The paper describes an algorithm for consistent business process decomposition and its applicability to the identification of business services.
- **Research rigour.** The artefact addresses a problem identified in the enterprise engineering and business process modelling literature. The solution is grounded on the principles of role modelling, separation of concerns and business process modelling.

- **Communication of research.** The research is reported through publications aimed at the practitioners and researchers within the enterprise engineering area and mainly targets business process modellers.

7 Conclusion and Future Work

Activity decomposition is an abstraction technique that enables the modularization of business processes. A decomposed process is easier to understand as each decomposition step incrementally reduces the number of overlapping concerns. This fosters the reuse and identification of the supporting services and increases the ability to communicate and analyse them. Each decomposition step provides a consistent level of detail so that the set of atomic activities comprising the lowest level of decomposition are always coherent, regardless of the stakeholder's requirements and the modelling team's experience.

The aim of the project is to guide the procedure of process decomposition so that decompositions are explicit and consistent. The proposed method supports the decomposition of business processes according to the separation of overlapping concerns. Business processes are modelled as the collaboration of natural types that play role types in the context of activities. A role ontology is used to specify the domain of role types and constrains the decomposition space. This approach facilitates the consistent decomposition of a process and the identification of the atomic activities, which contributes to service identification. However, the scenario presented in this paper does not evaluate the impact of the specification of the ontology and the overhead introduced by role-modelling in business process modelling. To overcome this limitation, we are currently developing a set of case studies intended to evaluate the applicability of the method to large-scale business processes.

References

1. N. Bieberstein, S. Bose, M. Fiammante, K. Jones, and R. Shah, *Service-Oriented Architecture (SOA) Compass: Business Value, Planning, and Enterprise Roadmap*. New York, NY, USA: IBM Press 2005.
2. M. Lankhorst, *Enterprise Architecture at Work: Modelling, Communication and Analysis*. Berlin/Heidelberg, Germany: Springer-Verlag, 2006.
3. T. Davenport, *Process Innovation: Reengineering Work Through Information Technology*. Boston, MA: Harvard Business School Press, 1993.
4. J. L. G. Dietz, *Enterprise Ontology: Theory and Methodology*. Berlin: Springer-Verlag, 2006.
5. M. Op't Land, E. Proper, M. Waage, J. Cloo, and C. Steghuis, *Enterprise Architecture: Creating Value by Informed Governance*. Heidelberg, Germany: Springer-Verlag, 2009.
6. P. Huber, K. Jensen, and R. M. Shapiro, "Hierarchies in Coloured Petri Nets," in *10th International Conference on Application and Theory of Petri Nets*: Springer, LNCS, vol. 483, 1990, pp. 313–341.

7. L. Bass, P. Clements, and R. Kazman, *Software Architecture in Practice*. Reading, Massachusetts: Addison-Wesley, 1998.
8. OMG, "Business Process Modeling Notation Specification. v 1.1 (formal/2008-01-17)." 2008.
9. A.-W. Scheer, *Business Process Modeling*, 3rd ed. Berlin: Springer Verlag, 2000.
10. R. J. Mayer, C. P. Menzel, M. K. Painter, P. S. deWitte, T. Blinn, and B. Perakath, *Information Integration for Concurrent Engineering - IDEF3*: Knowledge Based Systems Inc., 1995.
11. W. Reisig and G. Rozenberg, *Lectures on Petri Net*, 1st ed. vol. 1491. Heidelberg, Germany: Springer, 1998.
12. M. Kloppmann, D. Koenig, F. Leymann, G. Pfau, A. Rickayzen, C. v. Riegen, P. Schmidt, and I. Trickovic, "WS-BPEL Extension for Subprocesses BPEL-SPE," IBM and SAP Joint White Paper 2005.
13. R. Davis and E. Brabdänder, *ARIS Design Platform*. London, UK: Springer-Verlag, 2007.
14. J. E. Ingvaldsen and J. A. Gulla, "Model Based Business Process Mining," *Journal of Information Systems Management*, vol. 23, 2006.
15. P. Bolstorff and R. Rosenbaum, *Supply Chain Excellence: A Handbook for Dramatic Improvement Using the SCOR Model*, 2nd ed. Berlin, Germany: Springer, 2008.
16. APQC, "APQC Process Clarification Framework - Consumer Products, version 5.0.2, 10/04/2008," 2008.
17. T. Hornung, A. Koschmider, and G. Lausen, "Recommendation Based Process Modeling Support: Method and User Experience," in *27th International Conference on Conceptual Modeling (ER'08)*.
18. W. v. d. Aalst, H. Beer, B. v. Dongen, and B. Springer-Verlag, "Process Mining and Verification of Properties: An Approach based on Temporal Logic," in *OTM Confederated International Conferences*. vol. 3760, Springer, 2005, pp. 130-147.
19. W. v. d. Aalst, H.Reijers, A. Weijters, B. v. Dongen, A. A. d. Medeiros, M. Song, and H. Verbeek, "Business Process Mining: An Industrial Application," *Information Systems Journal*, vol. 32, pp. 713-732, 2007.
20. M. Uschold, M. King, S. Moralee, and Y. Zorgios, "The Enterprise Ontology," *The Knowledge Engineering Review*, vol. 13, pp. 31-89, 2000.
21. G. Greco, A. Guzzo, L. Pontieri, and D. Sacca, "An ontology-driven process modeling framework,," in *15th International Conference on Database and Expert Systems Applications*, F. Galindo, M. Takizawa, and R. Traunmuller, Eds. Zaragoza, Spain: IEEE Computer Society, 2004, pp. 13-23.
22. A. Albani, J. L. G. Dietz, and J. Zaha, "Identifying Business Components on the basis of an Enterprise Ontology," in *Interoperability of Enterprise Software and Applications*. Springer, 2006, pp. 335-347.
23. C. W. Bachman, "The role data model approach to data structures," in *International Conference on Databases*, S. M. Deen and P. Hammersley, Eds.: Heyden & Son, 1980, pp. 1-18.
24. B. Kristensen, "Object-Oriented Modeling with Roles," in *2nd International Conference on Object-Oriented Information Systems*, 1995.
25. T. Reenskaug, P. Wold, and O. Lehn, *Working With Objects: The OOram Software Engineering Method*. Greenwhich: Manning Publication Co., 1996.
26. F. Steimann, "On the representation of roles in object-oriented and conceptual modelling," *Data & Knowledge Engineering*, vol. 35, pp. 83-106, 2000.
27. D. Riehle, "Framework Design: A Role Modeling Approach." vol. PhD Zurich, Switzerland: Swiss Federal Institute of Technology, 2000, p. 229.
28. M. Ould, *Business Processes: Modeling and analysis for re-engineering and improvement*. Chichester: John Wiley & Sons, 1995.

29. J. Krogstie, S. Carlsen, A. Consulting, and I. L. Chicago, "An integrated modelling approach for process support," in *30th Hawaii International Conference onSystem Sciences, HICSS 1997*. vol. 2, 1997.
30. A. Wegmann, "On the systemic enterprise architecture methodology," in *International Conference on Enterprise Information Systems (ICEIS 2003)* Angers, France, 2003.
31. L.-S. Lê and A. Wegmann, "SeamCAD: Object-Oriented Modeling Tool for Hierarchical Systems in Enterprise Architecture," in *39th Hawaii International Conference on System Sciences*, Hawaii, USA, 2006.
32. M. Zacarias, A. Caetano, R. Magalhães, H. S. Pinto, and J. Tribolet, "Towards Organizational Self-Awareness" in *Ontologies for Business Interactions*, P. Rittgen, Ed.: Idea Group Inc., 2007.
33. A. Caetano, A. Rito Silva, and J. Tribolet, "A Role-Based Enterprise Architecture Framework," in *24th Annual ACM Symposium on Applied Computing, ACM SAC 2009*, Hawaii, USA, 2009.
34. M. Zacarias, R. Magalhães, H. S. Pinto, and J. Tribolet, "An agent-centric and 'context-aware' perspective for the alignment between individuals and organizations," *Information Systems (2009.03.014)*, 2009.
35. J. Sowa, *Conceptual Structures: Information Processing in Mind and Machine*. New York: Addison-Wesley, 1984.
36. N. Guarino, M. Carrara, and P. Giaretta, "An Ontology of Meta-Level Categories," *Principles of Knowledge Representation and Reasoning: Proceedings of the Fourth International Conference (KR94). Morgan Kaufmann, San Mateo, CA*, pp. 270-280, 1994.
37. S. Carlsen, "Comprehensible Business Process Models for Process Improvement and Process Support," in *8th International Conference, CAISE'96*. vol. 1080, Crete, Greece: Springer LNCS, 1996.
38. G. M. Giaglis, "A Taxonomy of Business Process Modeling and Information Systems Modeling Techniques," *International Journal of Flexible Manufacturing Systems*, vol. 13, pp. 209-228, 2001.
39. S. March and G. Smith, "Design and natural science research on information technology," *Decision Support Systems*, vol. 15, pp. 251-266, 1995.
40. A. R. Hevner, S. T. March, J. Park, and S. Ram, "Design science in information systems research," *MIS Quarterly*, vol. 28, pp. 75-105, March 2004 2004.
41. C. Braun, F. Wortmann, M. Hafner, and R. Winter, "Method construction - a core approach to organizational engineering," in *ACM Symposium on Applied Computing*, USA, 2005, pp. 1295-1299.