A Method for Business Process Decomposition based on the Separation of Concerns Principle

Artur Caetano
artur.caetano@ist.utl.pt
António Rito Silva
rito.silva@ist.utl.pt
José Tribolet
jose.tribolet@ist.utl.pt

Department of Computer Science and Engineering,
Instituto Superior Técnico, Technical University of Lisbon, 1049-001 Lisboa, Portugal.

Center for Organizational Design and Engineering,
INESC INOV, Rua Alves Redol 9, 1000-029 Lisboa, Portugal.

ABSTRACT

Functional decomposition breaks down a business process into a set of progressively more detailed activities. It facilitates the modular design of a system, the reuse of its parts and also contributes to increasing its comprehensibility. But achieving these qualities requires a business process to be decomposed consistently. Separation of concerns is the principle of separating a system into distinct features with a minimum of overlapping. This paper proposes using this principle to consistently decompose a business process into its constituent activities. An activity is modelled as a collaboration between role types that are played by entities. The decomposition method successively separates the overlapping roles until an activity is specified by the collaboration of an orthogonal set of role types. This method facilitates the consistent decomposition of a business process and the unambiguous identification of its atomic activities.

Categories and Subject Descriptors

D.2.2 [Design Techniques]: Object-oriented design methods
H.1.0 [Models and Principles]: General
I.6.5 [Simulation and Modelling]: Model development

Keywords

business process modelling, functional decomposition, role modelling, 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 representing it in the form of a knowledge base. This transformation is usually not straightforward as the source knowledge is not structured or formalized and tends to be of complex nature. In addition, the purpose of the model itself may not be well defined or even understood by its stakeholders. As a matter of fact, a number of researchers posit that complexity is an essential property of the design activities due to the complexity of completely formulating the problem and to the inability to cope simultaneously with all of its constraints [1].

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SAC 10, March 22-26, 2010, Sierre, Switzerland.
Copyright 2010 ACM 978-1-60558-638-0/10/03 ...$10.00.
different teams, probably with different backgrounds and experience, who are involved in the elicitation, modelling and analysis of an organization’s processes. Put together, these factors add to models that lack consistency. Examples of inconsistency include using different levels of modelling detail and modelling styles and the incoherent naming of the activities and the 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.

Consistent business process decomposition can significantly improve the clarity and the overall model integrity as well as minimizing the omission of relevant information [5]. 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 [6]. As a consequence of abstraction, models become easier to understand and communicate, which, in turn, make their validation, redesign and optimization easier.

The level of the functional decomposition of a process depends on the modeller and on the purpose and scope of the process model. Each decomposition level describes process elements from a different abstraction level. To do so, users have to satisfy specific modelling requirements, such as homogenous abstraction of process element names on the same decomposition level in order to improve model consistency.

This paper presents a method that specifies how to decompose a business process according to the concerns that are involved in the specification of its activities. A business process activity is constructed as a collaboration of roles played by entities. Each role abstracts the behaviour that an entity displays in the context of an activity. The set of roles played by an activity defines the process’ decomposition space and thus the superset of functional views that can be generated.

The remainder of this paper is structured as follows. Section 2 introduces the concepts of natural type, role type and activity. Sections 3 and 4 describe the functional decomposition method and the underlying role ontology along with a running example. Section 6 reviews related work and section 7 summarizes the research methodology behind this work. Finally, section 8 summarizes the proposal and provides an outlook on future work.

2. FUNDAMENTAL CONCEPTS

This section defines the foundations behind role-based business process modelling [7, 8]. An activity (business verb) is abstracted as a set of collaborations between entities (business nouns). The entities represent the things within the organization that are of interest in a specific modelling context. Each unit of behaviour that an entity displays is abstracted as a role.

Figure 1 shows the relationships and corresponding cardinalities between four entities involved in the assemble product business process which we will use to exemplify the concepts previously outlined. This activity describes how a product is assembled from a number of parts (two in this example) by means of an assembling machine. The activity is semi-automated since the machine is operated by a person.

Each of the relationships between the entities results in one collaboration context where a natural type displays a specific behaviour modelled with a role type (v. Figure 2).

In the collaboration context c1, each of the two parts plays the role of input resource in their relationship with the assembling machine which, in its turn, is playing the actor role. In the context c2, the assembling machine produces the assembled product, i.e., the product is the output resource of this actor. In the context c3, the person relates to the machine as an actor. The collaboration between these four roles uniquely defines the assemble product activity as depicted in Figure 3.

The remainder of this section defines the three concepts introduced along with this example, namely entity (natural type), role type and activity.

2.1 Natural Types and Role Types

Sowa [9] 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 et al. [10] presented an ontological distinction between these two types. A role type is founded and not semantically rigid. Role types require to always stand in some relation to other individuals (i.e. the type is founded), and they can enter and leave the type without losing their identity (i.e. the type is not semantically rigid). 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 Figure 3 and classify the concepts of person and actor as either natural types or role types. First, let us focus on the “foundedness” of these two concepts. Actor is a founded type since for something or someone to enter the actor type there must be something being acted upon or operated; i.e., an actor must always stands in relation to some subject. Conversely, the person type is not founded since it exists on its own terms regardless of being in a relationship with any other type. Thus, person is a not founded type and actor is a founded type.

Regarding the type’s “semantic rigidness”, the actor type is not semantically rigid because its identity is independent of en-
A business process involves specifying the set of activities that produces goods or provides services that add value to the organization. Therefore, an activity is a role type (founded and not semantically rigid) whereas a person is a natural type (not founded and semantically rigid).

2.1.1 Natural Types or Entities

Entities are natural types. An entity describes all things an organization deems relevant to store some information about for a specific purpose and in the context of a specific model. These include concepts such as persons, places, machines and products. According to the definition of natural type, an entity can always be unambiguously identified and defined in isolation, i.e. without any relationship with other type. Entities can be classified according to its intrinsic properties. Entities may relate structurally to other entities, as in the case an entity is composed by other entities.

2.1.2 Role Types

A role type, or role for short, is the observable behavioural 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 properties of that entity when it collaborates with another entity in the context of an activity. An entity relates to zero or more roles through the play relationship. An entity that plays no roles is not participating in any activity, i.e., it produces no behaviour. An entity enters the role type when it starts playing that role and leaves the role when that specific behaviour is concluded. Each role adds a set of external properties to an entity in the context of that collaboration.

Figure 4. Property augmentation.

Figure 4 depicts the entity Person playing two roles: Employee and Insurant. In this example, Person has three intrinsic properties (name, age, sex) that are part of its natural type. While a Person is playing the Employee role in collaboration with the Employer role its feature space is augmented by the job position and salary properties. A different and independent set of properties (e.g. the insurant’s contract number and the insurance policy) is brought in while the Person plays the Insurant role in a different context. These distinct sets of external attributes depend on the specific role being played by the entity.

This approach separates the property space of the entity: its intrinsic properties are differentiated from the external properties that transiently relate to entity through the roles it plays.

2.2 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. Modelling a business process involves specifying the set of activities that comprise it and the flow that defines how the activities are orchestrated. The same set of activities may be orchestrated differently in several processes. This means activity specification and activity orchestration are two orthogonal concerns.

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. A role type separates the description of the intrinsic properties of an entity from the properties that derive from the collaborations it participates in. Thus, the specification of an activity is not only independent from the orchestration it is involved in but also from the specification of the entities that play the roles pertaining to that activity.

Figure 3 depicts the assemble product activity as a unit of functionality that result from the collaboration between a set of roles. This activity’s 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 functional 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, meaning two activities are equivalent if and only if they share the same set of role collaborations.

3. 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. sections 1 and 6) 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 decompose a process using the role type collaborations as the criteria for decomposition. This means each decomposition step explicitly separates a concern from the other concerns that specify the activity. An activity is deemed atomic, meaning it cannot be further decomposed, when there are no more overlapping roles left in its specification, i.e. when all the similar concerns are separated. This also implies that the classification of an activity as atomic actually depends on the role ontology that is being utilized to build the model. So, different role ontologies set different functional decomposition criteria and, thus, yield different process models. The algorithm for activity decomposition can be applied to an activity at any level of detail as the resulting set of atomic activities is equivalent.

This algorithm consists of the function decompose(S, R). The set S contains the role type instances ri that specify the activity to be decomposed. The decomposition criteria are defined by the role types specified in the set R. This function recursively adds to the result set of non-overlapping roles for each level of decomposition. An activity is considered atomic, meaning it cannot be further decomposed according to the role ontology, whenever all of its role type instances are non-overlapping. The algorithm is detailed in Table 1.
Table 1. Algorithm for activity decomposition.

```
decompose(S, R)
    D ← Ø
    decompose'(S, R, D, 1)
    decompose ← D
end
decompose'(S, R, D, level)
    if R ≠ Ø then
        R₀ ← firstElementOf(R)
        D₀ ← Ø
        if numInstancesOfType(R₀, S) > 1 then
            for all r ∈ R₀ do
                Sₙ ← (S - R₀) ∪ r
                Dₙ ← D₀ ∪ Sₙ
                decompose'(Sₙ, R - R₀, D, level +1)
            end for
            else
                decompose'(S, R - R₀, D, level +1)
            end if
        end if
    end if
end
```

Figure 5 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 second 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 Figure 6.

Decomposing A1 according to the Locator role (R3) yields two activities, A1.1 and A1.2, as shown in Figure 7. 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 Figure 8 and Figure 9.


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.

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. Additionally, the condition for activity decomposition is explicit as the procedure stops whenever the concerns of an activity are effectively separated.
4. ROLE ONTOLOGY

The decomposition method relies on a role type ontology to model the collaborations taking place in the context of a business process. The ontology represents the set of role types required to model a specific domain and the relationships between these types.

A business process can be modelled from different perspectives according to the model’s goals and purpose. Although there are multiple classification schemes to categorize the modelling perspectives, we posit that these crosscut the six orthogonal linguistic interrogatives (how, what, where, who, when, why). These interrogatives can be used to create the following modelling perspectives [11, 12]:

- **Functional**: represents what activities are being performed in the context of a given process.
- **Informational**: represents what informational entities (i.e. data or resources) are being manipulated by the activities of a process.
- **Behavioural**: represents when activities are performed and how they are performed, usually through the specification of the process orchestration.
- **Organizational**: represents why an activity is being performed, where it is performed and by whom.

The remainder of this section illustrates a set of roles types that addresses the above concerns according to the six interrogatives.

4.1 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 Figure 3 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.

4.2 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 focuses 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.)

4.3 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.

4.4 Goal, Rule (Why)

A goal represents a measurable state of affairs that the organization intends to achieve. The entity that specifies such state of affairs 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 compiler role.

5. ACTIVITY ORCHESTRATION

Orchestration is a concern that captures how and when the activities within a process are sequenced. This is modelled by the specification of the constraints that limit how activities are ordered. These constraints include control flow constructs, such as the AND-split and the AND-join, which are widely covered by business process and workflow modelling languages [13, 14].

The ontology used to capture the activity orchestration concern makes use of the role types finisher and starter. The finisher role is played whenever the activity completes, i.e., when the collaboration between its roles comes to an end. Conversely, the starter role signals the start of the activity. These two roles capture the successor-predecessor relationships occurring during the process orchestration. Each control flow construct is therefore also modelled as an activity. This section describes the modelling of two constructs, sequence and AND-split, but the same reasoning here described also applies to other control flow constructs. A sequence or sequential flow defines a basic successor-predecessor constraint, i.e. an activity starts after the completion of another activity. The AND-split creates multiple successors by dividing the flow into multiple branches.

Figure 11 shows the role collaborations within the sequence and AND-split activities. Sequence is an activity defined by the collaboration between the finisher and starter roles. The starter role is played by an entity whenever the previous activity terminates its execution. This entity is usually a resource that has been generated by the activity or an event that signals its termination. As the sequence construct is unconditional, the starter role immediately follows the finisher role. The AND-split is mod-
elalled similarly, but it is the collaboration between a starter role with at least two finisher roles.

As a result, each control flow construct is modelled as an activity, which, in its turn, is modelled as a collaboration between starter and finisher role types. This means the same role-based principles are used for modelling the activities within a business process as well as the activities that specify the control flow.

![Activity orchestration diagram](image)

Figure 12. Activity orchestration.

Figure 12 shows the orchestration sequence(A1, A2), AND-split(A3, A4). The top part represents the specification of activities A1-A4. The bottom part represents the control flow specification, which includes two control flow activities: sequence and AND-split. The middle part of the figure depicts how activities A1-A4 are orchestrated using the sequence and AND-split constructs.

6. RELATED WORK

Functional decomposition is supported at language level by most process modelling languages, including ArchiMate [15], BPMN, [16] EPC [17] and IDEF-0/3 [18]. The decomposition of subsystems through the hierarchic classification of process models has also been applied to Petri nets [19] and BPEL [20]. Although these approaches make possible to create a hierarchical representation of a process, their intent is not the definition of specific methods for consistent activity decomposition but the representation of generic decomposition structures. Yet, the shortcomings of the lack of consistency in process decomposition and in the specification of atomic activities are pointed out by some authors [21, 22].

Various 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 [23]. The Process Clarification Framework defines a hierarchical decomposition of business processes which is 3-4 levels deep and crosses 12 operating and management categories [24]. Other approaches, such as the ARIS framework [17], 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 3-4 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 (represented with Petri nets) to detect inconsistencies between its activities [25]. 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. Other approach defines how to perform the functional decomposition of activities that are specified with AND/OR trees or production rules [26]. Process mining techniques extract information from existing event logs and enable the discovery of business processes [27]. 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 [28]. However, process mining is based on event logs and therefore assumes a process is already being supported by the organization’s information systems. Other approaches that use ontologies to specify business processes (e.g. [29-31]) 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 means to consistently decompose a business process or to unambiguously identify the atomic activities that constitute the process.

7. RESEARCH METHODOLOGY

The project reported in this paper follows a design science research approach, which focuses on the development of effective solutions for practical problems [32, 33]. Research on enterprise engineering fits this paradigm as its main goal is creating methods and techniques to analyze, model, and understand the interface between the organization and the supporting information systems and technology [34]. The tangible result of a design science project consists in creating an artefact that addresses a particular issue that is relevant to a group of stakeholders. Hevner et al. proposed a set of guidelines to conduct design science research projects [33]. The following list briefly summarize how a subset of these guidelines was applied to this work.

- **Design as an artefact.** This project deals with applying the principle of separation of concerns to enterprise architecture. The artefact presented in this paper describes process decomposition using the aforesaid principle.

- **Problem relevance.** The goal of this paper is to facilitate the consistent analysis and design of business processes. The problem is relevant as decomposition plays a major role in business process modelling.

- **Design evaluation.** The utility of the artefact is demonstrated using a small set of scenarios [33]. In the near future, we will focus on evaluating the artefact using real-world business processes.

- **Research rigour.** The artefact is based on a rigorous knowledge base that includes separation of concerns, abstraction strategies and business process modelling.

- **Communication of research.** The research is being reported through publications aimed at the practitioners and researchers within the enterprise engineering area.
8. 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 of the process models and increases the ability to communicate and analyze them. Each decomposition step must provide 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 specifies the domain of role types and constrains the possible decomposition space. This approach facilitates the consistent decomposition of activities and the identification of the atomic activities of a process according to a given role ontology. The methodology behind the development of this artefact is grounded on the design science research principles.

We are currently working on the specification of the role ontologies that translate a number of specific constructs derived from mainstream business process modelling languages and the development of case studies to assess the advantages and drawbacks of the method when applied to large-scale business processes.

REFERENCES