Tool Support for the Semantic Object Model

Otto K. Ferstl, Elmar J. Sinz and Dominik Bork

Abstract This chapter introduces tool support for the semantic object model (SOM). The conceptual design of a multi-view modeling tool is presented after describing the core concepts of the SOM method and laying the corresponding methodological foundation. The chapter foremost addresses the modeling enthusiast, interested in how to utilize the SOM method with the ADO*xx* modeling tool.

Keywords Semantic object model · Multi-view modeling tool · ADO*xx*

1 Introduction

The semantic object model (SOM)[1] is a comprehensive methodology for modeling business systems [1–4, 194ff]. SOM is fully object-oriented and designed to capture business semantics explicitly. The general bases of the SOM methodology are concepts of systems theory as well as organizational theory.

SOM supports the core phases of business engineering such as analysis, design and redesign of a business system. A business system is an open, goal-oriented, socio-technical system. Thus, the analysis of a business system focuses on the interaction with its environment, goal-pursuing business processes, and resources. Moreover, the dynamic behavior of a business system requires investigation of properties such as stability, flexibility and complexity [5].

[1]This section is based on [1].

O.K. Ferstl · E.J. Sinz (✉)
University of Bamberg, 96045 Bamberg, Germany
e-mail: elmar.sinz@uni-bamberg.de

O.K. Ferstl
e-mail: otto.ferstl@uni-bamberg.de

D. Bork
University of Vienna, 1090 Vienna, Austria
e-mail: dominik.bork@univie.ac.at

D. Karagiannis et al. (eds.), *Domain-Specific Conceptual Modeling*,
DOI 10.1007/978-3-319-39417-6_13

The backbone of the SOM methodology is an enterprise architecture which uses different perspectives on a business system via a set of models. These models are grouped into three model layers referred to as business plan, business process models and resource models. Each layer describes the business system as a whole, but with respect to the specific perspective on the model. In order to reduce complexity, each model layer is subdivided into several views, each focusing on specific aspects of a model layer. On the meta level, the modeling language of each layer is defined by a metamodel and derivated view definitions. Thus, the enterprise architecture provides a modeling framework which helps to define the specific semantics and to manage the complexity of the model [6]. In this chapter, we outline the methodological framework of SOM, its modeling language as well as the conceptualization of an SOM modeling tool based on the ADO*xx* metamodeling platform.

In terms of systems theory, a business system is an open, goal-oriented, socio-technical system [7]. It is open because it interacts with customers, suppliers, and other business partners transferring goods and services. The business system and its goods/services are part of a value chain which in general comprises several consecutive business systems. A corresponding flow of finance runs opposite the flow of goods and services.

The behavior of a business system is aimed at business goals and objectives. Goals specify the goods and services to be provided by the system. Objectives (e.g., profit and turnover) are defined levels against which business performance can be measured.

Actors of a business system are humans and machines. Human actors are persons in different roles. Machine actors, in general, are plants, production machines, vehicles, computer systems, etc. SOM pays specific attention to application systems which are the machine actors of the information processing subsystem of a business system (information system). An application system consists of computer and communication systems running application software. The degree of automation of an information system is the ratio of tasks carried out by application systems to all tasks of the information system.

The notion of a business system as open and goal-oriented reflects a perspective from outside the system. An inside perspective shows a distributed system of autonomous, loosely coupled components which cooperate in pursuing the system's goals and objectives. The autonomous components are business processes [4, 8] which produce goods and services and deliver them to other business processes.

The cooperation of business processes is coordinated primarily through process-specific objectives which are derived from the overall objectives of a business system. This is done by the business system's management. Within the degrees of freedom defined by the process-specific objectives, a secondary coordination is done by negotiation between the business processes.

Inside a business process, there are components which also cooperate and have to be coordinated. This coordination is done by an intra-process management which controls the activities of the process components by sending instructions to them and supervising their behavior. In contrast to the coordination between business

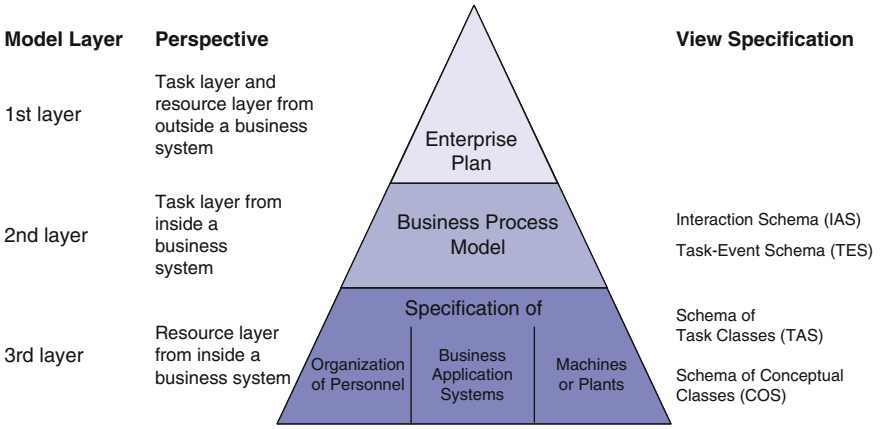

Fig. 1 Enterprise architecture [4]

processes, the components of a business process are guided closely by the process management.

The components of a business process as well as the business processes as a whole take care of functions which are essential to every business system. The following classification of these functions helps to identify business processes and their components: (1) input-output-function to implement the characteristic of openness, e.g., a production system, (2) supply function to provide material resources and energy, (3) maintenance function to keep the system running, (4) sensory function to register disturbances or defects inside or outside the system, (5) managing function to coordinate the subsystems [9].

2 The SOM Enterprise Modeling Method

The SOM methodology[2] utilizes an enterprise architecture which consists of three layers (Fig. 1) [4].

- Enterprise plan: The enterprise plan constitutes a perspective from outside a business system. It focuses on the global task and the resources of the business system. The specification of the global task includes the universe of discourse, the goals and objectives to be pursued, as well as the goods and services to be delivered. Requirements on resources are derived from the global task and have to be cross-checked to the capabilities of available resources. Hence, both global task and resources determine themselves mutually.

[2]This section is based on [1].

The first evaluation of an enterprise plan is done by an analysis of chances and risks from a perspective outside the business system, and an additional analysis of the strengths and weaknesses of the business system from an inside perspective. Strategies on products and markets, strategic actions, constraints, and rules serve as guidelines to realize an enterprise plan.

- Business process model: The business process model constitutes a perspective from inside a business system on its tasks. It specifies main processes and service processes which consist of tasks and relationships between them. Main processes contribute directly to the goals of the business system; supporting processes provide their outcome to main processes or other supporting processes. The relationships between business processes follow the client/server concept. A client process engages other processes for delivering the required service. Business processes establish a distributed system of autonomous components. They cooperate in pursuing joint objectives which are derived from the overall objectives of a business system.
- Specification of resources: In general, personnel, application systems as well as machines or plants are resources for carrying out the tasks of business processes. In the following, we focus on information processing tasks and, therefore, omit machines and plants. Tasks of the information system are assigned to persons or to application systems classifying a task as non-automated or fully-automated. A partly-automated task has to be split into sub-tasks which are non-automated or fully-automated. The assignment of persons or application systems is aimed at the optimal synergy of person–computer cooperation.

The different layers of the enterprise architecture help to build business systems in a flexible and manageable way. They cover specific aspects of an overall model which are outside perspective (enterprise plan), inside perspective (business process model) and resources. The relationships between the layers are specified explicitly. Each layer establishes a distributed system of autonomous, loosely coupled components. In contrast to a single-layered monolithic model, the multi-layered system of three models allows local changes without affecting the overall architecture. For example, it is possible to improve a business process model (inside perspective) yet retain goals and objectives (outside perspective) or replace actors of one type by other ones.

Following an outside-in approach, it is advisable to build the three model layers top down the enterprise architecture. However, the architecture does not force this direction. There may be good reasons to start from this guideline, e.g., when analyzing existing business systems. Here it is sometimes difficult to find an elaborated enterprise plan so modeling starts at the business process layer focusing on the inside perspective. The enterprise plan may be completed when the other layers are fully understood. In each case, the effects on other layers have to be balanced and approved.

The enterprise architecture implies that functionality and architecture of the business application systems are derived from the business process model. The relationships between both layers are formaliszd to a high degree. Design decisions

and results at the business process layer are translated automatically into the layer of application systems. The architecture of the layer of application systems uses the concept of object-integration to combine conceptual and task classes [10]. Alternatively, it is possible to link a business process model to an existing, traditional application system which follows the traditional concepts of function integration or data integration. In this case, tasks to be automated are linked to functional units of the application system.

3 Conceptualization of the SOM Modeling Method

In this section, the language for SOM business process[3] models is defined. The language is specified by a metamodel (Sect. 3.1) and a set of decomposition rules (Sect. 3.2). Finally, Sect. 3.3 briefly introduces SOM resource modeling.

3.1 The Metamodel for Business Process Modeling

The metamodel for business process modeling shows notions and relationships between notions (Fig. 2). It is specified as a binary entity-relationship schema. Relationships between notions are associated with a role name as well as two cardinalities to denote how many instances of the one notion can be connected to one instance of the other notion, at least and at most. Within the metamodel, the notions are represented by entities. Each entity also contains the symbols used for representation within a business process model.

As introduced in Sect. 2, a business process model specifies a set of business processes with client/server relationships among each other. A business process pursues its own goals and objectives which are prescribed and tuned by the management of a business system. Cooperation between processes is a matter of negotiation. The term business process denotes a compound building block within a business process model and, therefore, it is not a basic notion of the language. A business process consists of at least one business object and one or more business transactions.

At the initial level of a business process model, a business object (object in short) produces goods and services and delivers them to customer business processes. Each business object belongs exclusively to a business process of the universe of discourse or to the environment of a business system. A business transaction (transaction in short) transmits a good or service to a customer business process or receives a good or service from a supplier business process. A transaction connecting different business processes belongs to both processes.

[3]This section is based on [1].

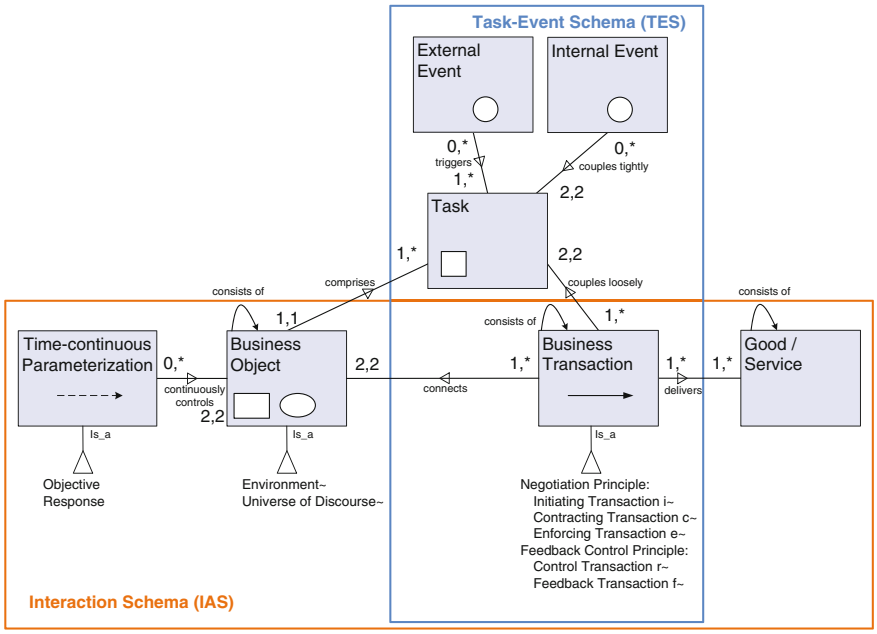

Fig. 2 Metamodel for SOM business process models [1, p. 344; 11, p. 219]

A business process may be refined using the decomposition rules given below in Sect. 3.2. At a more detailed level of a business process model, each business object appears in one of two different roles: an operational object contributes directly to producing and delivering a good/service, while a management object contributes to managing one or more operational objects using messages. A business transaction transmits a good/service or a message between two operational objects or a message between two management objects or between a management object and an operational object.

A business transaction connects two business objects. Conversely, a business object is connected with one to many ("*") in-going or out-going business transactions. From a structural viewpoint, a transaction denotes an interaction channel forwarding goods, services, or messages. From a behavioral viewpoint, a transaction means an event which is associated with the transmission of a specific good, a service package, or a message.

A business object comprises one to many tasks, each of them driving one to many transactions. A transaction is driven by exactly two tasks belonging to different business objects. The tasks of an object share common states and are encapsulated by the object. These tasks pursue joint goals and objectives which are attributes of the tasks.

The SOM methodology uses two different concepts of coupling tasks (Fig. 3, top): loosely coupled tasks belong to different objects and, therefore, operate in

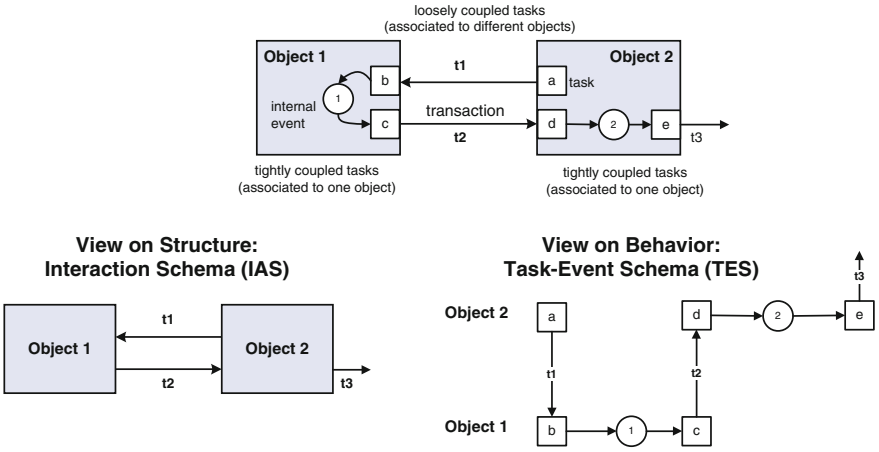

Fig. 3 Representation of structure and behavior in an SOM business process model

different states. The tasks are connected by a transaction which serves as an interaction channel for passing states from one task to the other. A task triggers the execution of another task by an event (good, service package, or message) riding on the interaction channel. Tightly coupled tasks belong to the same object and operate on the same states. The tasks are connected by an internal event which is sent from one task to trigger the execution of the other. The concept of encapsulating tightly coupled tasks by an object and loosely coupling the tasks of different objects via transactions is a key feature of the object-oriented characteristic of the SOM methodology.

The third type of event is the external event. An external event denotes the occurrence of an event like "the first day of a month" which is not bound to a transaction.

Because of its complexity, a business process model is represented in two different views (Fig. 3 bottom and Fig. 2): The interaction schema is the view of the structure. It shows business objects which are connected by business transactions. The task-event schema is the view on behavior. It shows tasks which are connected by events (transactions, internal events, or external events). These two views are complemented by hierarchical decompositions of business transactions and business objects. These additional views specify the relationships between the interaction schemas showing a varying level of detail.

3.2 Decomposition Rules

The SOM methodology allows a business process model to be decomposed by stepwise refinement. Decomposition takes place with the components of the

interaction schema specifying the structure of a business process model, i.e., business objects, business transactions and goods/services (see the relationship "consists of" in Fig. 2). The components of the task-event schema which specify the behavior of a business process model (tasks, events riding on transactions, internal events and external events) are not decomposed but redefined on subsequent decomposition levels of a business process model. The decomposition rules for business objects and business transactions are shown in Fig. 4. Specific rules for decomposition of goods/services are not required because of simply decomposing them into sub-goods/sub-services.

The decomposition of a business process model helps to manage its complexity, allows separating the management system of a business process from its operational system, and uncovers the coordination of a business process.

The SOM methodology uses two basic coordination principles within decomposition [4]:

- Applying the feedback control principle (rule 1), a business object is decomposed into two sub-objects and two transactions: a management object O' and an operational object O'' as well as a control transaction T_r from O' to O'' and a feedback transaction T_f in opposite direction. These components establish a feedback control loop. The management object prescribes objectives or sends control messages to the operational object via the control transaction. Conversely, the operational object reports to the management object via the feedback transaction.

Rule Nr.	*Decomposition rules for business objects:*	
(1)	O	::= { O', O'', T_r(O', O''), [T_f(O'', O')] }
(2)	O	::= { O', O'', [T(O', O'')] }
(3)	O	::= { spec O' }$^+$
(4)	O' | O''	::= O
(5)	O	::= { O', { O'', P_o(O', O''), [P_R(O'', O')] }$^+$ }
	Decomposition rules for business transactions:	
(6)	T(O, O')	::= [[T_i(O,O') seq] T_c(O', O) seq] T_e(O, O')
(7)	T_x	::= T'_x { seq T''_x }$^+$ | T'_x { par T''_x }$^+$ (x = i, c, e, r, f)
(8)	T_x	::= { spec T'_x }$^+$ (x = i, c, e, r, f)
(9)	T_i | T_c | T_e	::= T

Fig. 4 Decomposition rules for business objects and business transactions (::=replacement, { } set, { }+list of repeated elements, [] option, | alternative, seq sequential order, par parallel order, spec specialization)

- Applying the negotiation principle (rule 5), a transaction is decomposed into three successive transactions: (1) an initiating transaction T_i, where a server object and its client learn to know each other and exchange information on deliverable goods/services, (2) a contracting transaction T_c, where both objects agree to a contract on the delivery of goods/services, and (3) an enforcing transaction T_e, where the objects transfer the goods/services.

The types of transactions resulting from the decomposition are shown in the metamodel (Fig. 2) as specialized transactions.

Figure 5 illustrates the application of the coordination principles for the decomposition of SOM business process models. The decomposition of the first level into the second level is done by applying the negotiation principle. Applying the feedback control principle leads to the third level.

In addition to the coordination principles given above, a transaction may be decomposed into sub-transactions of the same type which are executed in sequence or in parallel (rule 6). Correspondingly, a business object may be decomposed into sub-objects of the same type (management object or operational object) which may be connected by transactions (rule 2). Objects, as well as transactions, may be specialized within the same type (rules 3 and 7). The other rules (4, 8, and 9) are used for replacement within successive decompositions.

It is important to state that successive decomposition levels of a business process model do not establish new, different models. They belong to exactly one model and are subject to the consistency rules defined in the metamodel.

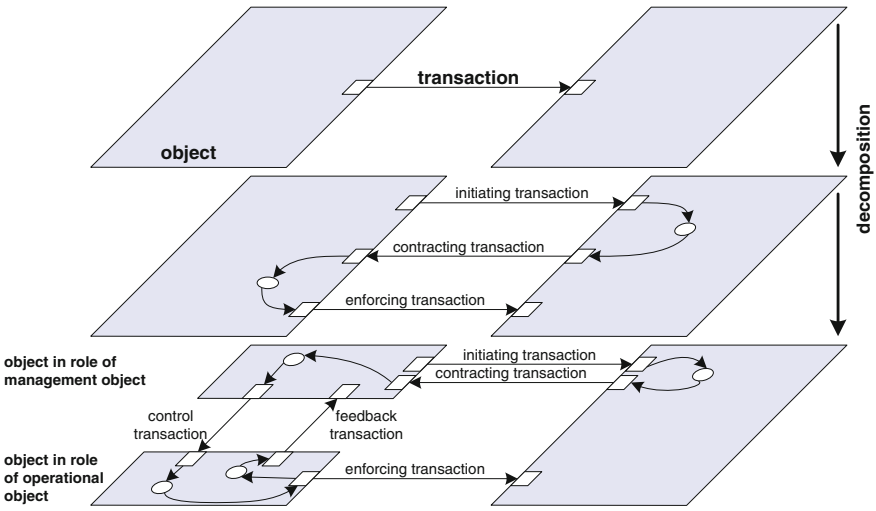

Fig. 5 Decomposition of SOM business process models

3.3 Resource Modeling

Tasks which are fully or partly automated and hence are executed by application systems have to be specified on the resource layer. The creation of the corresponding schemas is initially done by a model-driven approach. As this approach cannot be described here in detail, only a short synopsis is given. For a detailed description see [11].

Following the organizational foundation of the SOM method, the focus is on the concept of task. Tasks at the business process layer are embraced by means of a common object. These objects are modeled by (object-specific) classes at the resource layer. Transactions between objects are mapped to (transaction-specific) classes as well as relationships between classes. Both types of classes result in a schema of conceptual classes (COS). The tasks at the business process layer are modeled by (task-specific) classes at the resource layer, leading to the schema of task classes (TAS).

A task-specific class orchestrates the corresponding sub-schema of conceptual classes. Task-specific classes at the same level are coordinated by choreography.

4 The SOM Modeling Tool

This section discusses the conceptualization of the theoretical foundation introduced above towards a modeling tool for the SOM method. The tool is based on the ADO*xx* metamodeling platform. Section 4.1 denotes SOM multi-view modeling functionality. Thereafter, Sect. 4.2 briefly discusses model transformation capabilities, enabling the derivation of resource layer models from business process models with SOM. Section 4.3 then denotes non-functional requirements.

4.1 SOM Multi-view Modeling Functionality

The SOM tool enables modeling of SOM business process models and resource models (cf. level 2 and level 3 of the SOM enterprise architecture on Fig. 1). On the different levels, SOM utilizes different ways of carrying out multi-view modeling. On the business process layer, all views are created following the *system-oriented multi-view modeling* approach [12]. Hence, all views are projections onto the integrated business process metamodel (cf. Fig. 2). Modeling actions, performed by the modeler on one view are immediately transformed by *transition translations* [13] into corresponding changes on all affected views.

On the resource level, both modeling views are kept isolated from each other. Modeling on this layer follows the *diagram-oriented multi-view modeling* approach [12]. However, considering the creation of the resource views, the SOM method specifies *state translations* [13] that transform complete business process models into semantically equivalent schema of task classes and schema of conceptual class models. These state translations are specified as metamodel mappings. Notions of the business process metamodel are mapped to notions of the resource layer metamodel (see Sect. 4.2 for the metamodel mapping and an example model transformation).

The SOM tool is special when it comes to the way modelers interact with the tool. Generally, conventional drag and drop modeling is prohibited at most times. This comes not only from the way multi-view modeling is performed, it is also a requirement that comes from the formalized specification of the modeling procedure [cf. 14] by means of the decomposition rules (cf. Fig. 4). In order to increase the utility of the tool, zooming operators were introduced, allowing modellers to immediately switch between already defined decomposition levels of a business process. Applying the zooming operator causes only changes on the visualized SOM views. The integrated model is not affected. Table 1 provides an overview of the constituents of the multiple SOM modeling views (realized as ADO*xx* model types) and aligns the most important tool functionality to them.

Table 1 SOM views realized as ADO*xx* model types and corresponding tool functionality

SOM view (ADO*xx* model type)	Modeling concept	Tool functionality
Interaction schema	Business object	Increase business process level
	Business transaction	Decrease business process level
		Auto-layout/smooth edges
Task-event schema	Task	Define process behavior
	Business transaction-internal/external event	Increase business process level
		Decrease business process level
Object decomposition schema	Business object	Decompose object/Transaction
	Business transaction	revoke decomposition
Transaction decomposition schema	Decomposition relationship	Zoom on selected level
		Add/Remove environmental object
		Add/Remove enforcing transaction

4.2 SOM Model Transformation Functionality

The SOM methodology not only specifies the business process and resource modeling on the second and third layer of the enterprise architecture, respectively. Moreover, SOM also defines a metamodel-based model transformation of comprehensively specified business process models into initial business application systems models (i.e., the schema of task classes and the schema of conceptual classes).

The SOM modeling tool provides the modeler the functionality to automatically apply these transformations. The hereby created business application systems models can be further processed with the tool, e.g., combining classes that have significant functionality and/or data overlaps, normalization or generalization of classes.

Figure 6 illustrates the metamodel mapping between the business process layer and the resource layer (cf. the SOM enterprise architecture in Fig. 1). Tasks are transformed into task-specific classes (rule 1), business objects into object-specific classes (rule 2), business transactions into transaction-specific classes and interacts_with relationships (rule 3), goods/services are transformed into service-specific classes, and internal/external events are transformed into interacts_with relationships.

In Fig. 7, this mapping is exemplified by a simple SOM business process, consisting of a distributor and a customer that are coordinated by three business transactions. This model is transformed into a schema of conceptual classes (Fig. 7 bottom left) and a schema of task classes for the customer business object (Fig. 7 bottom

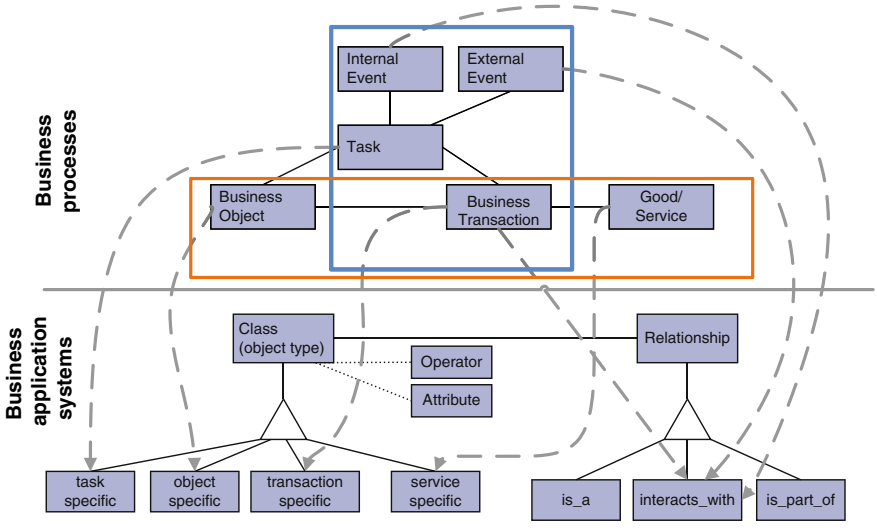

Fig. 6 Transforming SOM business processes into SOM business application systems

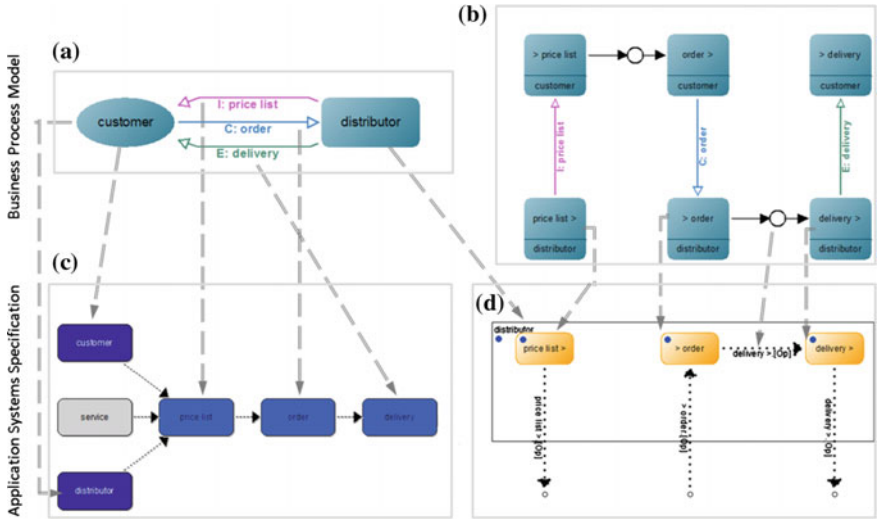

Fig. 7 Interaction schema (**a**), task-event schema (**b**) and the transformed schema of conceptual classes (**c**) and schema of task classes (**d**)

right). The dashed arrows indicate some of the transformation rules applied to the business process model. Because of the limited space available here, the resource layer and the corresponding transformation rules cannot be discussed in more detail. However, the interested reader is referred to [11, p. 222ff] for an overarching specification of both, the resource layer of SOM and the transformation rules.

4.3 Non-functional Tool Requirements

From the very beginning, the conceptual design of the SOM modeling tool was aiming at the best usability, i.e., how to support the modeler in processing multiple views simultaneously and applying the decomposition rules. In [15], the authors emphasized the importance of:

- Decoupling the decomposition of business objects and business transactions from the definition of new business process levels by means of specifying the relationships between the decomposed objects and transactions;
- providing visual support in the reconfiguration of relationships between business objects in different decomposition levels;
- enabling zooming in and out of already defined business process levels; and
- realizing layout algorithms, e.g., "auto-layout" and "smooth-edges" that automatically adjust the visualization of the SOM business process model in its multiple views after a modeling action has been executed.

Consistency is a major requirement for multi-view modeling tools. This holds all the more for SOM modeling. Because of the multiple modeling views on the different enterprise architecture layers, modelers are confronted with many dependencies to be respected. Hence, automatic mechanisms for ensuring consistency need to be provided on tool side.

The SOM modeling tool utilizes the ADO*xx* event mechanisms to realize this requirement. Every modeling operation is checked in an according event processing algorithm. This algorithm determines whether or not the current modeling operation, performed in a certain view, affects other views. If so, the algorithm executes semantically equivalent operations (referred to as transition translations [13]) at all affected views automatically. Hence, without interrupting or confusing the modeler, consistency between all SOM views is provided automatically by the tool.

5 Case Study: A Product Distribution Process in SOM

In the following section, the application of the SOM modeling tool is demonstrated by means of a case study, showing the different modeling steps to be applied in order to transform an initial SOM business process into a precise description of a distribution of goods/services between a distributor and a customer. One focus of the SOM method is on business process modeling. Because of the limited space available here, the following case study will, therefore, concentrate on modeling of SOM business processes.

For example, Fig. 8 introduces the business process distribution of a trading company visualized as a screen shot of the SOM modeling tool. At the initial level, the interaction schema view (bottom left model type) consists of three components, (1) the business object distributor which provides a service, (2) the transaction service which delivers the service to the customer (visualized in the transaction decomposition view on the upper right), and (3) the business object customer itself. Distributor is an internal object belonging to the universe of discourse while customer is an external object belonging to the environment. All business objects are visualized in the upper right view, the object decomposition view. At this level, the entire cooperation and coordination between the two business objects is specified by the transaction service.

Figure 8 (bottom right) shows the corresponding sequence of tasks which is very simple. The task names in the task-event schema are derived from the name of the transaction. Here, the task *service* > (say "send service") of distributor produces and delivers the service, the task > *service* (say "receive service") of customer receives it. The arrow service here defines the sequence of the two tasks belonging to the transaction service which is represented in the interaction schema by an arrow, too.

Transactions like services connect business objects inside the universe of discourse and link business objects to the environment. When modeling a value chain, the business process model of a trading company includes a second business

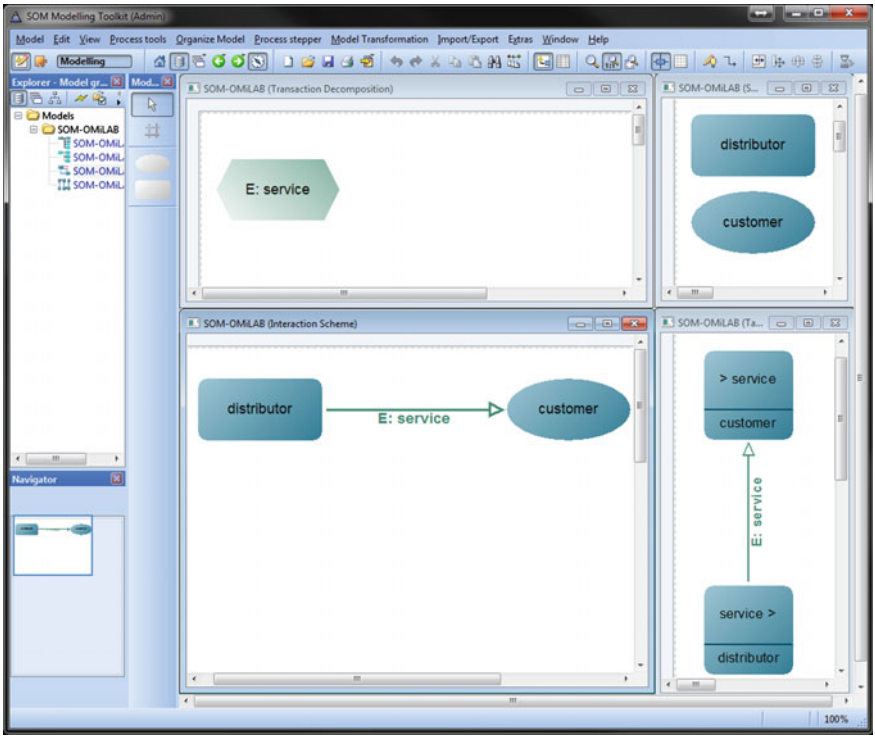

Fig. 8 Initial level of the business process distribution process

process procurement, which receives services from a business object supplier, belonging to the environment, and delivers services to a distributor.

Example (Fig. 8) will be continued now. For readability reasons, the figures concentrate on selected views in the following. The surrounding text will describe the modeling steps performed in all views. As customer and distributor negotiate about the delivery of a service, the service transaction is decomposed according to the *negotiation principle* into the sub-transactions *i: price list* (initiating), c: *order* (contracting), and e: *service* (enforcing transaction); visualized in the transaction decomposition schema in Fig. 9a. The corresponding task-event schema (Fig. 9d) is determined implicitly because the sub-transactions are executed in sequence (as defined by the negotiation decomposition rule in Fig. 4). The tasks of each business object are connected by object-internal events.

After this initial step, the resulting business transactions and business objects need to be further decomposed to more precisely depict the actual distribution of goods and services:

First, the e: service transaction is decomposed into the sequence *e:(seq.) delivery* and *e:(seq.) cash up*. The cash up transaction is further decomposed according to the *negotiation principle* into the sequence *c: invoice* and *e: payment* (see Fig. 10a).

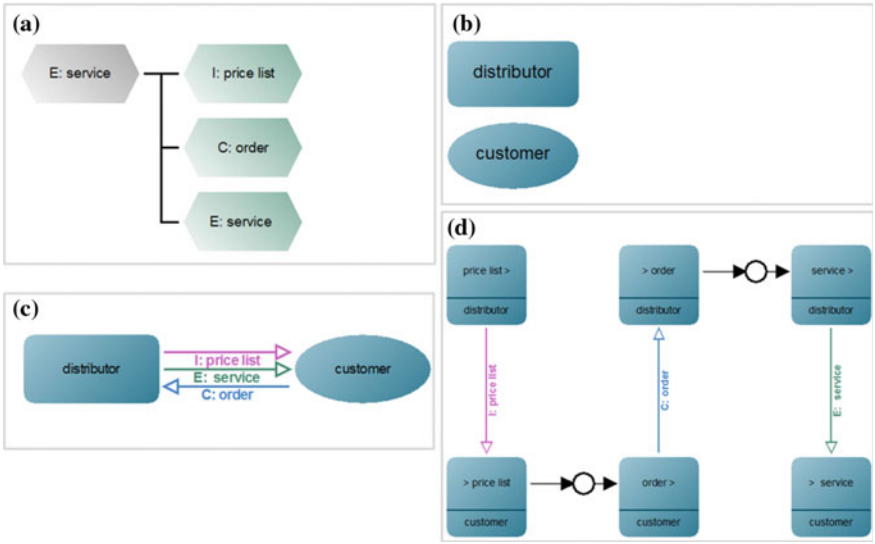

Fig. 9 Transaction decomposition schema (**a**), object decomposition schema (**b**), interaction schema (**c**), and task-event schema (**d**) on the 2nd level

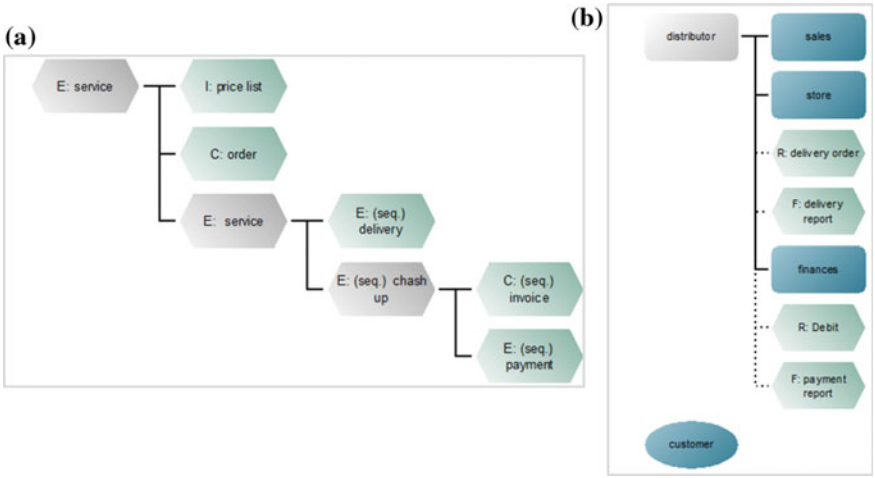

Fig. 10 Transaction decomposition schema (**a**) and object decomposition schema (**b**) on the 3rd level

The initiating transaction is omitted because the business objects already know each other. The contract of the invoice transaction refers to amount and date of payment, not to the obligation to pay in principle which is part of the transaction c: order.

Second, the feedback control principle is applied two times to distributor to (i) uncover the internal management of the business object, and (ii) derive at a homogeneous mapping between business transactions and business objects. Following the feedback control principle (cf. Fig. 4), this leads to the sub-objects sales (management object), *store* (operational object) and *finances* (operational object). Sales and store are coordinated by the transactions r: *delivery order* (control transaction) and f: *delivery report* (feedback transaction). Sales and finances are coordinated by the transactions r: *debit* and f: *payment report* (see Fig. 10b).

Figure 11 shows the interaction schema of the third business process level. The sales sub-object deals with *price list, (seq.) invoice* and *order,* the store sub-object is responsible for delivery. Consequently, the finances sub-object takes care of the *(seq.) payment*.

The last step in this case study is to define the behavior of the business process on its final decomposition level. Because of the different decomposition rules applied, the sequence of transactions cannot be derived automatically for the final process. Hence, the modeler is required to define the process behavior in the task-event schema by utilizing the internal event relationship and consecutively clicking on the outgoing and incoming task an internal event shall connect. Figure 12 shows the final task-event schema.

The business process is still initiated by the sales object sending a price list to the customer. The customer then sends an order back to sales. This initiates a control transaction delivery order to the store that actually delivers the good or service to the customer and responds with a feedback transaction (delivery report). After the report is processed, the sales object initiates two transactions: The sales object sends an invoice to the customer and it requests a debit to be handled by the finances. On receiving the customer's payment, the finances reports by means of a feedback transaction the payment report back to sales. This concludes the distribution process.

The complete case study with illustrations of all decomposition levels and views is available online at the Open Models Laboratory (OMiLAB) project page of

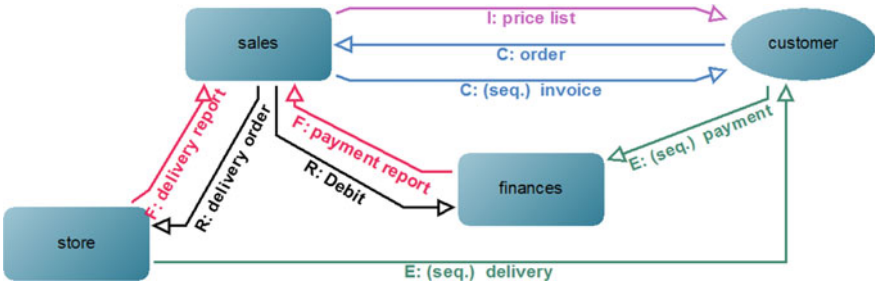

Fig. 11 Interaction schema on the 3rd level

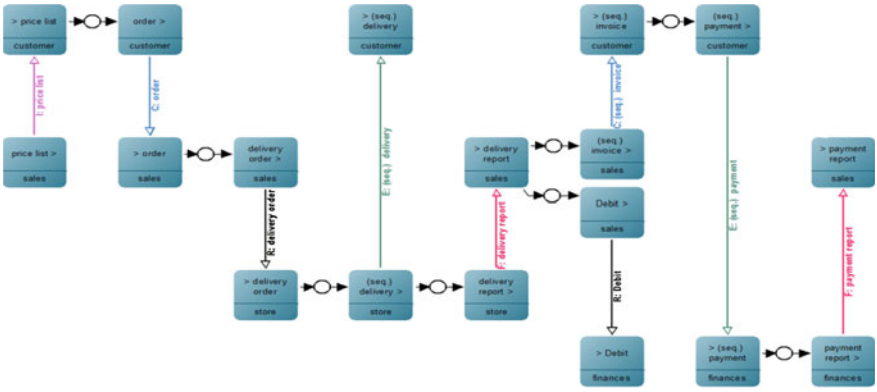

Fig. 12 Task-event schema on the 3rd level

SOM.[4] It also features the transformation of the final distribution business process into initial (i) schema of task classes, (ii) schema of conceptual classes and (iii) business process modeling and notation (BPMN) (cf. [16]). A comprehensive discussion on the conceptualization of the SOM method towards a multi-view modeling tool can be found in [17].

6 Conclusion

The previous sections gave a brief introduction to the SOM methodology for business systems modeling. A comprehensive enterprise model consists of sub-models for each layer of the enterprise architecture (see Fig. 1). The sub-models are balanced carefully within the architectural framework.

Consequently, the conceptualization of these characteristics towards the ADO*xx*-based SOM modeling tool was described. An emphasis was on the transformation of the multiple views of the method into model types and the general way of carrying out multi-view modeling with the tool.

Finally, method and tool were utilized in a case study illustrating the coordination of a distributor and a customer in a product or service distribution process.

Modeling, according to the SOM method, is a very knowledge-intensive endeavour. The sequence of modeling actions the modeler performs significantly influences the resulting business process model. Hence, modelers may face at some point that decisions taken (e.g., the chosen decomposition principle or the way the business transactions are connected to sub-objects after decomposition) end up requiring a revision. The SOM modeling tool, therefore, provides undo-operators

[4]The complete distribution business process case study, http://www.omilab.org/web/som/tutorial, [online] last access: 23.10.2015.

for almost any modeling action. With "revoke decomposition", the modeler is enabled to discard an already performed decomposition completely from all views in order to revise the decomposition. Moreover, relationships between business objects and business transactions can be revised independently from the decomposition itself by applying the "decrease business process level" operator (cf. Sect. 4.3).

Using the modeling tool in university courses at the University of Bamberg revealed that besides solid knowledge on the theoretical foundation of the SOM method, an introduction of the modeling tool is also required. Once students have this knowledge, the feedback gained after using the tool for solving modeling case studies was throughout positive. Future work will, therefore, focus on extending the SOM modeling tool project page[5] within the OMiLAB homepage with further tutorials, videos, a handbook and further FAQ's answered by the developers. The SOM modeling tool is freely available on the OMiLAB webpage.[6]

Tool Download http://www.omilab.org/som.

References

1. Ferstl, O.K., Sinz, E.J.: Modeling of business systems using SOM. In: Bernus, P., Mertins, K., Schmidt, G. (eds.) Handbook on Architectures of Information Systems. International Handbook on Information Systems, vol. 1, pp. 347−367, 2nd edn. Springer (2005)
2. Ferstl, O.K., Sinz, E.J.: Objektmodellierung betrieblicher Informationssysteme im Semantischen Objektmodell (SOM). WIRTSCHAFTSINFORMATIK **32**(6), 566–581 (1990)
3. Ferstl, O.K., Sinz, E.J.: Ein Vorgehensmodell zur Objektmodellierung betrieblicher Informationssysteme im Semantischen Objektmodell (SOM). WIRTSCHAFTSINFORMATIK **33**(6), 477–491 (1991)
4. Ferstl, O.K., Sinz, E.J.: Der Ansatz des Semantischen Objektmodells (SOM) zur Modellierung von Geschäftsprozessen. WIRTSCHAFTSINFORMATIK **37**(3), 209–220 (1995)
5. Bahrami, H.: The emerging flexible organization: perspectives from silicon valley. Calif. Manage. Rev. 33–52 (1992)
6. Sinz, E.J.: Architektur betrieblicher Informationssysteme. In: Rechenberg, P., Pomberger, G. (eds.) Informatik-Handbuch, pp. 875–887. Hanser-Verlag, München (1997)
7. Ferstl, O.K., Sinz, E.J.: Grundlagen der Wirtschaftsinformatik. Band 1, 3rd edn. Oldenbourg, München (1998)
8. Ferstl, O.K., Sinz, E.J.: Geschäftsprozeßmodellierung. WIRTSCHAFTSINFORMATIK **35**(6) 589–592 (1993)
9. Beer, S.: The Brain of the Firm, 2nd edn. Wiley, Chichester (1981)
10. Ferstl, O.K.: Integrationskonzepte betrieblicher Anwendungssysteme. Fachbericht Informatik 1/92. Universität Koblenz-Landau (1992)
11. Ferstl, O.K., Sinz, E.J.: Grundlagen der Wirtschaftsinformatik, Band 1, 7th edn. Oldenbourg, München (2013)

[5]OMiLAB project page of SOM, http://www.omilab.org/web/som, [online] last access: 23.10. 2015.

[6]Download the SOM modeling tool, http://www.omilab.org/web/som/download, [online] last access: 23.10.2015.

12. Bork, D., Sinz, E.J.: Bridging the gap from a multi-view modelling method to the design of a multi-view modeling tool. Enterp. Model. Inf. Syst. Archit. (EMISA) **8**(2) 25–41 (2013)
13. Bork, D., Buchmann, R., Karagiannis, D.: Preserving multi-view consistency in diagrammatic knowledge representation. In: Zhang, et al. (eds.) Proceedings of the 8th International Conference on Knowledge Science, Engineering and Management. LNAI vol. 9403, pp. 1–6 (2015)
14. Bork, D., Fill, H.-G.: Formal aspects of enterprise modeling methods: a comparison framework. In: Sprague, R.H.J. (ed.) Proceedings of the 47th Hawaii International Conference on System Sciences, pp. 3400–3409. IEEE Computer Society Press, Big Island, Hawaii, USA (2014)
15. Bork, D., Sinz, E.J.: Design of a SOM business process modelling tool based on the ADOxx meta-modelling platform. In: Lara, et al. (eds.) Pre-Proceedings of the 4th International Workshop on Graph-Based Tools, University of Twente, Enschede, pp. 90–101 (2010)
16. Puetz, C., Sinz, E.J.: Model-driven derivation of BPMN workflow schemata from SOM business process models. In: Enterp. Model. Inf. Syst. Archit. **5**(2), 57–72 (2010)
17. Bork, D.: A development method for the conceptual design of multi-view modeling tools with an emphasis on consistency requirements. Ph.D. thesis, University of Bamberg (2015)