BAMBERGER

BEITRÄGE ZUR WIRTSCHAFTSINFORMATIK

ISSN 0937-3349

Nr. 43

Modeling of Business Systems Using the Semantic Object Model (SOM) - A Methodological Framework -

Otto K. Ferstl, Elmar J. Sinz

July 1997

Accepted for: P. Bernus, K. Mertins, and G. Schmidt (ed.): Handbook on Architectures of Information Systems. International Handbook on Information Systems, edited by Bernus P., Blazewicz J., Schmidt G., and Shaw M., Volume I, Springer 1997

OTTO-FRIEDRICH-UNIVERSITÄT BAMBERG

TABLE OF CONTENTS

MODELING OF BUSINESS SYSTEMS USING THE SEMANTIC OBJECT MODEL (SOM) - A METHODOLOGICAL FRAMEWORK -

Otto K. Ferstl, Elmar J. Sinz[1]

1 Introduction

A business system which produces and delivers products and services is highly complex in general and requires valuable resources. It is reasonable to use experienced methods of business engineering instead of trial and error to build such a system and to ensure that the design goals can be achieved to a high degree. Business engineering allows to look at a business system as a whole. It accompanies its complete life cycle starting with the first beginning of business systems planning.

The Semantic Object Model (SOM) [FeSi90, FeSi91, FeSi95] is a comprehensive and integrated methodology for business engineering. It supports sound modeling of business systems and can be used for analysis and design.

SOM is based on concepts of systems theory. A business system is understood as an open, goal-oriented, and socio-technical system. It uses the coordination principles feedback control and negotiation. A business system is a dynamic system which requires investigation of problems like stability, flexibility, and complexity [Bahr92].

The backbone of the SOM methodology is an enterprise architecture which allows to divide a comprehensive and complex enterprise model into the model layers enterprise plan, business process model, and resources, each of them describing a business system completely and from a specific viewpoint. On the meta level, each layer is defined by a meta model as well as cor-

[1]{Univ.-Prof. Dr. Otto K. Ferstl, Univ.-Prof. Dr. Elmar J. Sinz}, Information Systems, University of Bamberg, D-96045 Bamberg, Germany. Phone ++49 951 863 {2679, 2512}, Fax ++49 951 863 {2680, 2513}, e-mail {otto.ferstl, elmar.sinz}@sowi.uni-bamberg.de.

responding view definitions and patterns. This enterprise architecture helps to manage complexity, specifies semantics and allows complete and sound modeling.

Business engineering using the SOM methodology is done from the perspective of business as a whole. This comprises enterprise plans and business processes as well as personnel and application systems which are resources to carry out business processes. A reduction of business engineering to the development of application systems impedes potential and significant results of organizational improvement. The following paper outlines the methodological framework of SOM and explains the metaphors behind its modeling concept.

2 Business systems

2.1 Characteristics of business systems

In terms of systems theory a business system is an open, goal-oriented, and socio-technical system [FeSi94]. It is open because it interacts with customers, suppliers, and other business partners transferring services and products. The business system and its products and services are part of a comprehensive value chain which usually comprises several companies. The companies transform and assemble material and products, gradually making more complex products from step to step. A corresponding flow of finance runs opposite the flow of products and services.

The behavior of a business system is aimed at business goals and objectives. Business goals specify the products and services to be provided by the business system. Objectives like profit and turnover determine to which extent the goals have to be pursued. The attainable extent also depends on the interactions between the companies along the value chain. These interactions are a matter of negotiation.

Actors of the tasks of a socio-technical business system are persons and machines. In particular, the machine actors of the information processing subsystem of a business system (i.e. the information system) are computers and communications systems. The degree of automation of an information system is the ratio of tasks carried out by computers to all tasks of the information system. The relationships between the different actors depend on how the tasks are assigned to them. A task assigned to a person who uses a computer defines a person-tool rela-

tionship where the person is responsible for the results of the tasks. Tasks which are assigned to persons and computers cooperating define partner-partner relationships.

The structural complexity and the behavioral complexity of a business system depend on several factors. First of them is the complexity of the environment of the business system and the complexity of the interface between business system and environment. A complex and turbulent environment means complex customer and supplier markets as well as strong competitors. It claims for a high variety of the systems behavior which becomes visible in a complex range of products and services with many variants and a large spectrum of delivery procedures. To implement a high variety of a business system's behavior, a highly complex structure of the system is needed. Indicators of the complexity of the structure are the number and the variety of the system's components and of the relationships between them. Relationships are used to deliver products and services between the components and to coordinate the delivering procedures.

2.2 Components of Business Systems

The notion of a business system as open and goal-oriented refers to an outside view on the system. The inside view shows a distributed system consisting of autonomous and loosely coupled components which cooperate in pursuing the system's goals. The autonomous components are business processes [FeSi93, FeSi95] which produce products and services and deliver them to the customers of the business system or to each other, acting in many respects like the business system itself.

Formally a business process consists of a business object and business transactions. The business object represents the producing part and is exclusively dedicated to the business process. The business transactions deliver products and services to customer processes or receive items from supplier processes. Transactions are shared by neighboring processes. 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. There is no need for a business process which has global control on the overall system.

In general there is an additional process management inside a business process which coordinates and controls the activities of the process by sending instructions to the process components and by supervising their behavior. In contrast to the principle of negotiation between

business processes, the components within a business process in general are guided closely by the process management.

From the viewpoint of systems theory there are basic subsystems within a business system which have to be designed compatible with the structure of business processes. The subsystems take care of functions which are essential in every business system and may also be identified within a business process. Basic subsystems are (1) an input-output-system, e.g. a production system to implement the characteristic of openness, (2) a supply system to provide material resources and energy, (3) a maintenance system for keeping the system running, (4) a sensory system to register disturbances or defects inside or outside the system, (5) a managing subsystem to coordinate the other subsystems [Beer81].

Viewing business systems as distributed systems of business processes introduces basic structural assumptions. Structural complexity has to be handled at two layers: (1) Building a business system first is a problem of selecting and combining the appropriate business processes. (2) Inside the business processes the required behavior is implemented by choosing the right managing and servicing procedures. At both layers it is possible to reuse configuration plans which have been tested in different business systems.

A business system can be evaluated using different criteria. In the context of this topic, flexibility and complexity are most important. A business system needs enough flexibility to adapt to changes of the environment and it should have minimum complexity in order to reduce costs of resources and management.

3 Basics of modeling business systems

3.1 Enterprise architecture

Critical issues of modeling a system are (1) to mark off the system to be modeled in a universe of discourse and (2) to specify the goals and objectives of modeling. Modeling goals declare the result of the modeling process. Modeling objectives relate to the result of the modeling process or to the process itself. They describe what is called a “good model“ and cover aspects like how to preserve the structure and behavior of a real system within its model, at the same time reducing the complexity of the model by abstraction. They also specify how to perform an efficient modeling process. Moreover, they tell how to manage complexity. Here, an a pri-

ori comprehensive model is split into several layers, each of them modeling the whole system under a specific viewpoint. This results in an architectural framework for business systems, which is called enterprise architecture (fig. 1).

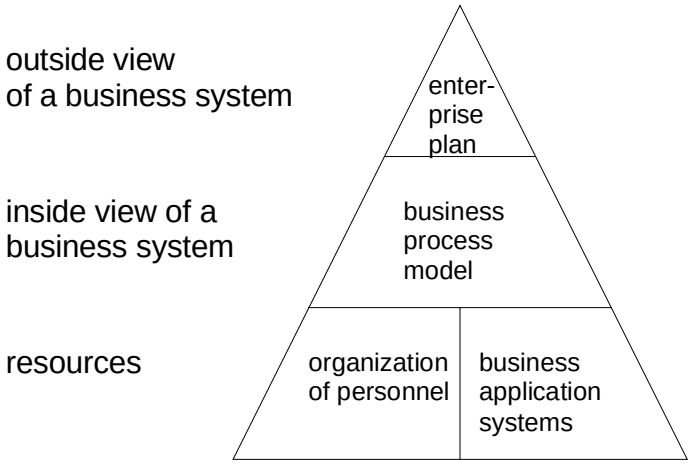

Fig. 1: Enterprise architecture [FeSi95]

The **enterprise architecture** consists of three layers [FeSi95]:

- **Enterprise plan**: The enterprise plan constitutes an outside view on a business system and focuses on its global task and required resources. The global task has to be marked off and specified. The specification includes the universe of discourse, the goals and objectives to be pursued, and the products and services to be delivered. Requirements on resources are derived from the global task and have to meet the capabilities of available resources. So both parts, global task and resources, are determined mutually. The business system and its environment is called the object system.

  A first evaluation of an enterprise plan is done by an analysis of chances and risks from a view outside the business system, and an additional analysis of the strengths and weaknesses of the business system from a view inside the system. Strategies on products and markets, strategic actions, constraints, and rules serve as guidelines to realize an enterprise plan. They aim at the interaction between the business system and its environment by planning the behavior of the system from a global perspective.

- **Business process model**: The business process model constitutes an inside view on a business system. It specifies main and service processes. Main processes contribute directly to the goals of the business system, service processes provide their outcome to other main or service processes. The relationships between business processes follow the client/server

concept. A client process engages other processes for delivering the required service. Business processes are autonomous components which cooperate in pursuing joint goals and which establish a distributed system.

- **Specification of resources**: Personnel and application systems are resources for carrying out the tasks of business processes. Within the enterprise architecture the resources are considered to be independent. So it is assumed that tasks can be assigned either to persons or to application systems classifying a task as non-automated or automated in full. A task automated in part can be split into sub-tasks which are non-automated or automated in full. The relationships between persons and application systems are considered to be partner-partner relationships. These rules of assignment allow to substitute application systems for persons and vice versa to get the most of the synergy of person-computer cooperation.

The different layers of the enterprise architecture help to build flexible and manageable business systems. They cover specific aspects of an overall model which are outside view (enterprise plan), inside view (business process model), and resources. The relationships between the layers are specified explicitly. Each layer employs autonomous and loosely coupled components.

Each model of a business system has to be balanced within and between the layers of the enterprise architecture. In contrast to a single-layered monolithic model, the multi-layered system of three models allows local changes without affecting the overall system. For example, it is possible to improve a business process model (inside view) yet retaining goals and objectives (outside view), additionally replacing actors of one type by other ones.

3.2 Generic Architectural Framework

The underlying methodical concept of the enterprise architecture is a **generic architectural framework** shown in fig. 2 [Sinz97]. In general, the architectural framework divides a complex model into several model layers, each of them specified according to a corresponding meta-model. To manage complexity, convenient views are defined on each model layer. Moreover, each layer is related to a particular set of design patterns, providing modeling rules and heuristic knowledge applicable to this model layer. The concept of design patterns applies the idea of object-oriented design pattern (see e.g. [Gamm+95]) to each model layer of the architectural framework. Relationships between different model layers are specified by rela-

tionship meta-models. Relationship patterns may be provided to support the connection of different model layers with transformation rules and other knowledge.

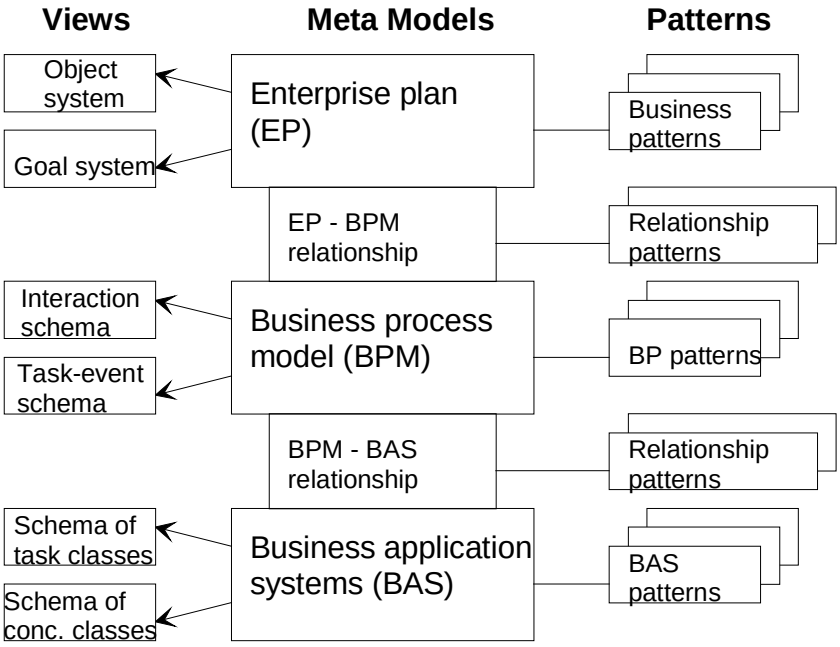

Fig. 2: Architectural framework for business system modeling [Sinz97]

In fig. 2 the architectural framework is instantiated with the domain-related layers of the SOM methodology. The three model layers match the enterprise architecture in fig. 1. The third layer is limited to application systems. Tasks assigned to persons are not considered within this layer. In the following chapters 4 and 5 we will refer to some specific components of the layers of business process model and business application systems.

3.3 The modeling procedure

Following an outside-in approach it is advisable to build the models of the three layers top down the enterprise architecture. Yet the architecture does not force this direction. Particularly there may be good reasons to depart from this guideline when analyzing existing business systems. Here it is sometimes difficult to find an elaborated enterprise plan, so modeling is started at the business process layer focusing on the inside view. The enterprise plan may be completed when the other layers are fully understood.

The enterprise architecture implies that the functionality and the architecture of the business application systems is derived from the business process model. The relationships between both layers are formalized to a high degree. Results and design decisions at the business proc-

ess layer are translated automatically into the layer of application systems. The architecture of this layer uses the concept of object-integration of conceptual and task classes which is compatible with the architecture of the business process model as with the concept of distributed objects in future application systems [Fers92].

The explicit differentiation between the two layers also allows to link a business process model to an existing and traditional application system following the concept of function integration or data integration although the relationships are not formalized. In this case the relationships link tasks of the process model, suitable for automation, to functional units of the application system, to realize an appropriate degree of automation.

The enterprise architecture is aimed at the results of the design process of a business system. It does not prescribe any specific sequence of design steps. The design steps and their sequence depend on the events which cause a model generation or a model modification. A modification in the enterprise plan may have an effect on the layers 2 and 3, whereas a modification of the business process model may influence layers 1 and 3. It also may happen that an enterprise plan and existing application systems are inputs for the design of a business process model. All effects on other layers have to be balanced and approved.

Since the enterprise architecture is not just a procedural model for building application systems but a blueprint of a business system, it has to be kept valid for the complete life cycle of a business system. Maintenance of the business system and its parts like application systems have to be done in accordance with the blueprint.

4 Business processes

4.1 Characteristics of Business Processes

In literature and practice, there are several approaches to business process modeling. Most of them assume a business process to be a sequence of activities (also called steps, process elements, functions), which are tied together by joint marks and which have to be equipped with resources [FeSi93, VoBe96]. In contrast, the SOM methodology to business process modeling considers three different constituents of a business process with the following characteristics [FeSi95]:

1. **Production and delivery** of one or more kinds of service (the term service comprises products, services and payments),
2. **Coordination of the business objects** which are involved in the production and delivery of these services, and
3. **Sequence of tasks** which are to be carried out when performing a business process.

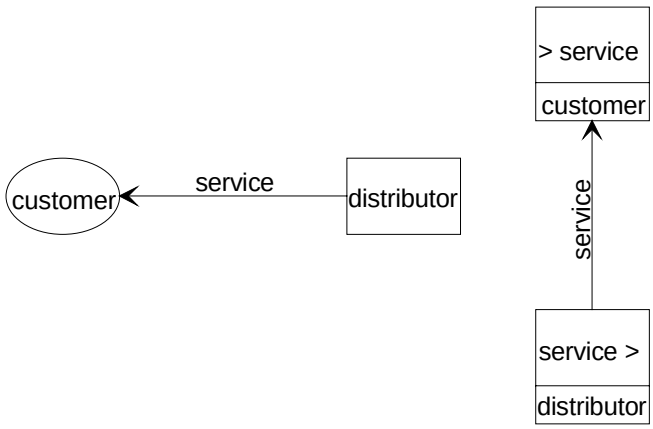

Fig. 3: Interaction schema (left) and task-event schema (right) of business process *distribution* (1st level)

The first two characteristics are represented in a diagram called **interaction schema**, the third one in a **task-event schema**. To give an example, figure 3 (left) introduces the business process *distribution* of a trading company. At the initial level, the interaction schema consists of three components, (1) the business object *distributor* which provides a service, (2) the transaction *service* which delivers the service to the customer, 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. At this level the entire coordination between the two business objects is specified by the transaction *service*. Figure 3 (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 *service* 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 process *procurement*, which receives serv-

ices from a business object *supplier*, belonging to the environment, and delivers services to *distributor*.

4.2 Metaphor of Business Process Modeling

The ideas behind the SOM methodology may be explained by metaphors which take views from outside and inside the business system. From the outside viewpoint of an enterprise plan, a business system is considered to be an open, goal-oriented, and socio-technical system. A business process model constitutes a corresponding inside view on a business system which is illustrated by a second metaphor. The inside view of a business system is linked to the outside view by the idea that a business process model specifies a procedure to carry out the enterprise plan. There may be several business process models that meet a given enterprise plan.

The SOM methodology understands a business process model as a distributed system [Ens78]. The model consists of autonomous components which encapsulate states and operations and which are loosely coupled among each other. The components send and receive service packages and messages to and from other components of the universe of discourse and its environment. A combination of two different paradigms is used to meet these characteristics: the components of a business process are specified according to the object-oriented paradigm, and the coordination of the components follows the transaction-oriented paradigm.

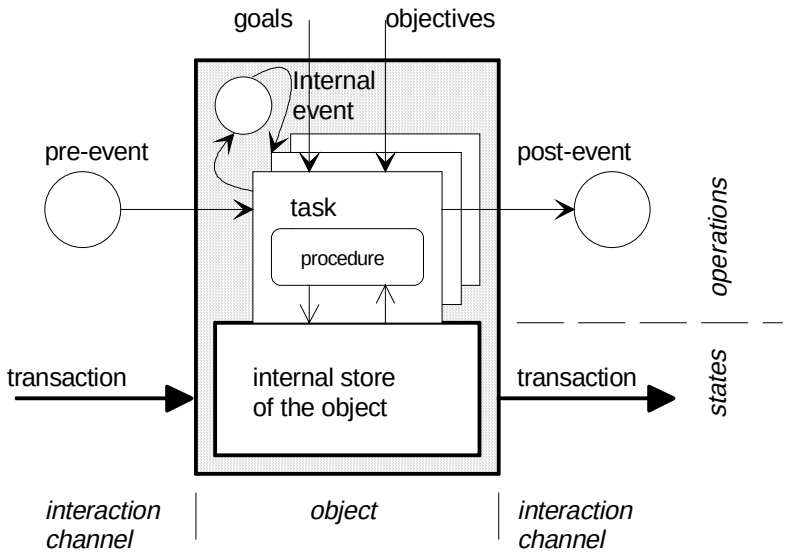

Fig. 4: The object-oriented concept of business objects

Figure 4 illustrates the **object-oriented concept** of business objects. A business object comprises a set of tasks pursuing joint goals and objectives and operating their procedures on common states. The states are part of the internal store of a business object. The internal stores of different business objects are connected by transactions which serve as interaction channels forwarding service packages and messages. Each service package or message is associated with an event which triggers the execution of a task. Tasks belonging to the same business object are connected by internal events.

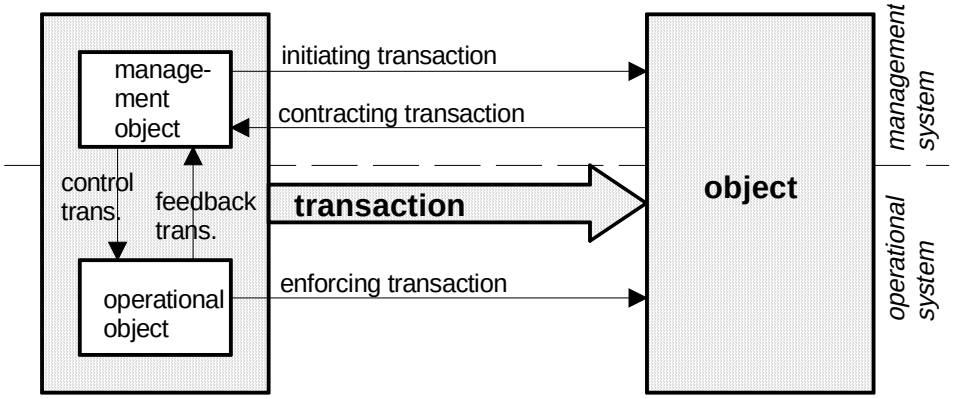

Fig. 5: The transaction-oriented concept of coordinating loosely coupled business objects

Figure 5 shows the **transaction-oriented concept** of coordinating loosely coupled business objects. The shadowed components in figure 5 denote a server and a client business object connected by a transaction. This extract from a business process model focuses on the operational part of a business system. To uncover the coordination between these business objects, the transaction and the business objects have to be decomposed. Doing this, the management system is separated from the operational system and becomes explicit. Two basic coordination principles are used within decomposition [FeSi95]:

- Following the **negotiation principle**, the transaction is decomposed into three successive transactions: (1) an initiating transaction, where the server and its client learn to know each other and exchange information on deliverable services, (2) a contracting transaction, where both objects agree to a contract on the delivery of services, and (3) an enforcing transaction, where the objects exchange the services.

- Applying the **feedback control principle**, a business object is decomposed into two subobjects and two transactions: a controlling and a servicing object as well as a control and a feedback transaction. These components establish a feedback control loop. The controlling

object prescribes objectives or sends control messages to the servicing object via the control transaction. Conversely the servicing object reports to the controlling object via the feedback transaction.

The transaction-oriented and the object-oriented concept complement each other. They are linked by the concept of tasks. A business object consists of a set of tasks, and each transaction is performed by exactly two tasks of different objects.

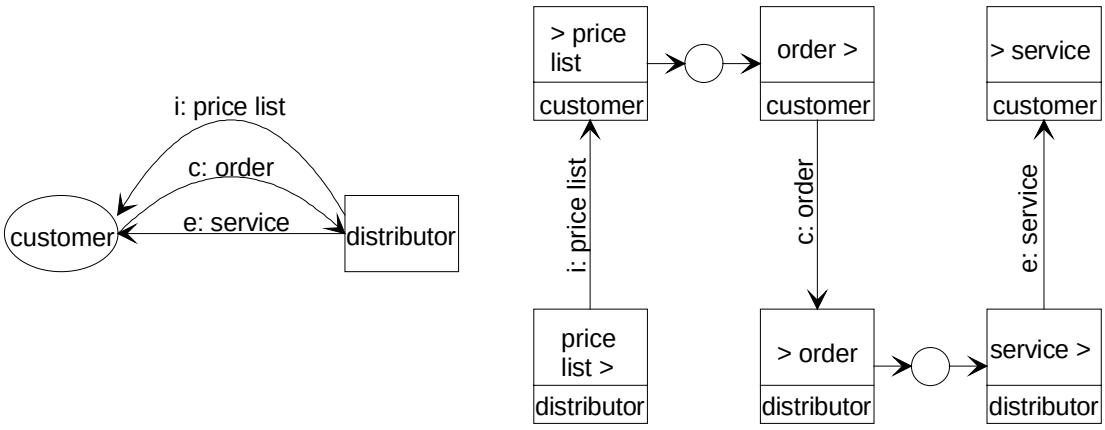

Fig. 6: Interaction schema (left) and task-event schema (right) of business process *distribution* (2nd level)

The example (fig. 3) will be continued now. 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). The corresponding task-event schema is determined implicitly because the sub-transactions are executed in sequence (fig. 6). The tasks of each business object are connected by object-internal events.

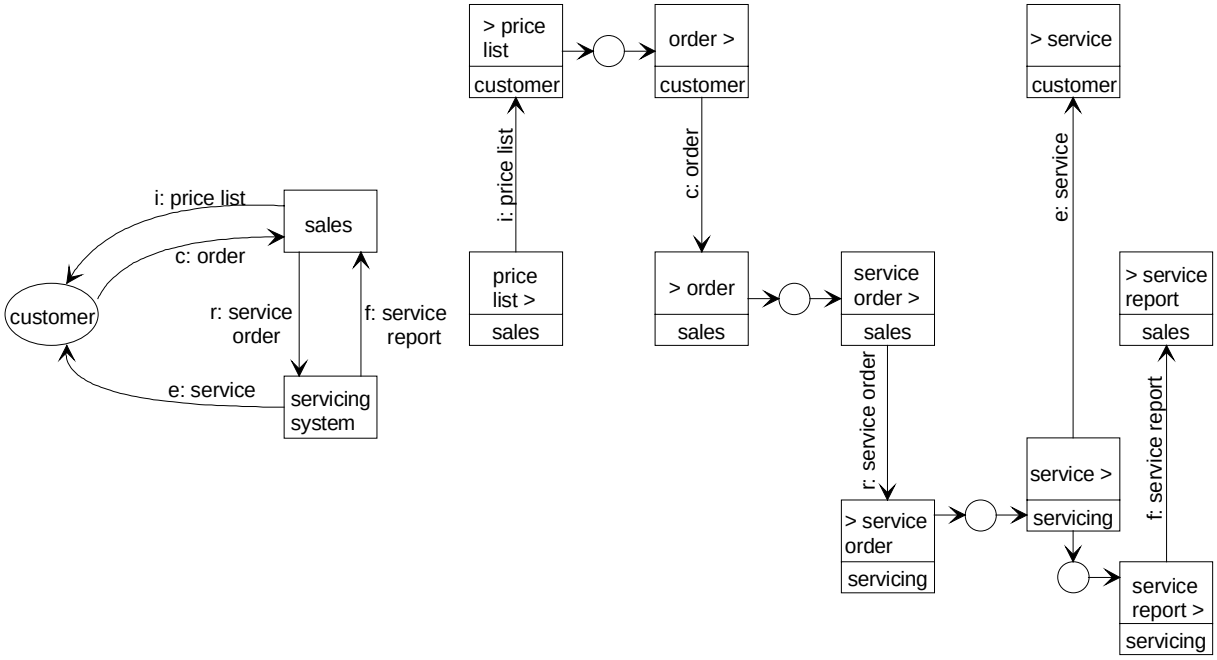

Fig. 7: Interaction schema (left) and task-event schema (right) of business process *distribution* (3rd level)

In the next step, the feedback control principle is applied to *distributor* to uncover the internal management of the business object. This leads to the sub-objects *sales* (controlling object) and *servicing system* (controlled object) as well as the transactions *r: service order* (controlling transaction) and *f: service report* (feedback transaction). At the same time the transactions assigned to the parent object *distributor* are re-assigned to the new sub-objects. The *sales* sub-object deals with *price list* and *order*, the *servicing system* operates the *service transaction* (fig. 7).

4.3 Meta Model for Business Process Modeling

The notions of business object, transaction, service, and event have been introduced in the previous sections. To specify their semantics and to enable complete and sound modeling [Sinz96], the notions are arranged to a meta model (fig. 8). Relationships between notions are associated with two cardinalities to denote how many instances of the one concept are connected to one instance of the other concept at least and at most.

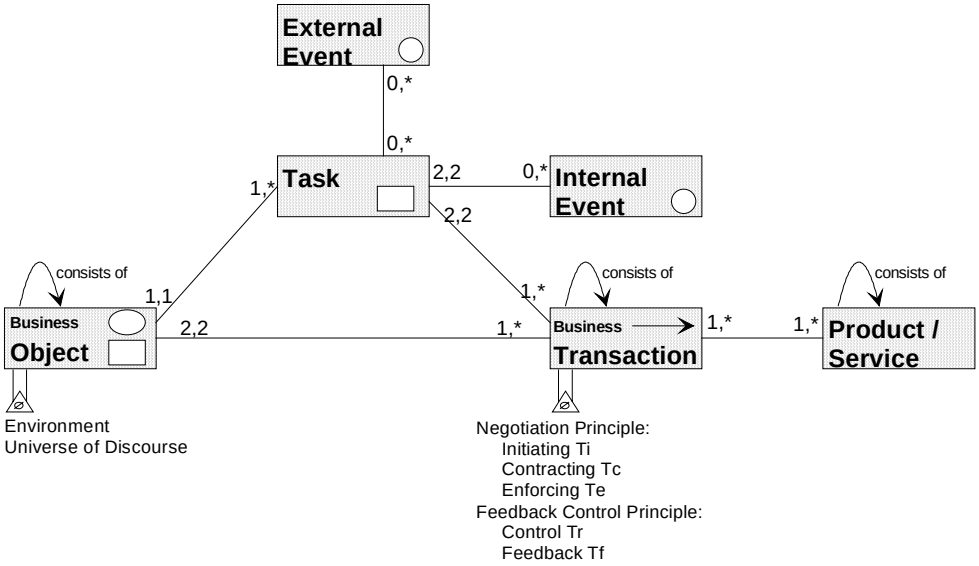

Fig. 8: Meta Model for Business Process Modeling [FeSi95]

The notion of business object is specialized into two sub-types. An object either belongs to the universe of discourse or to the environment. Each business object is connected with one to many („*“) in-going or out-going transactions. Transactions may be specialized according to the principles of negotiation or feedback control. Conversely, a transaction belongs to exactly two business objects. 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. Successive tasks within a business object are connected by internal events. External events denote the occurrence of events like „the first day of a month“ which are not linked to a transaction. Finally a transaction is associated with one to many products or services, and vice versa.

As introduced in the previous section, transactions and business objects may be decomposed according to the negotiation principle and the feedback control principle. In addition, a transaction may be decomposed into sub-transactions of the same type which are executed in sequence or in parallel. Correspondingly, a business object may be decomposed into sub-objects of the same type (controlling or controlled object) which may be connected by transactions. Services may be decomposed as well to uncover sub-services.

It is important to state that different decomposition levels of a model do not establish new, different models. They belong to the same model and they are subject to the consistency rules defined in the meta model.

Continuing the example, the final decomposition of the business process *distribution* uses the additional rules given above (fig. 9 and 10). Here, the *servicing system* and the *service* transaction are decomposed to find business objects and transactions which operate homogeneous products or services. First, the *e: service* transaction is decomposed into the sequence *e: delivery* and *e: cash up*. The *cash up* transaction is decomposed again according to the negotiation principle into the sequence *c: invoice* and *e: payment*. 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*.

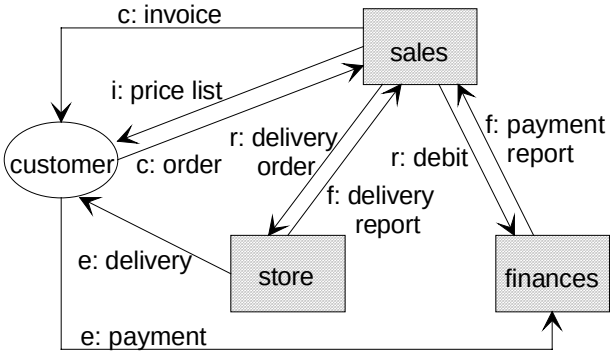

Fig. 9: Interaction schema of business process *distribution* (4th level)

As a result of this refinement, some other decomposition are necessary. The business object *servicing system* is decomposed into *store* and *finances*, responsible for products and payments respectively. The transaction *r: service order* is decomposed into the parallel transactions *r: delivery order* and *r: debit.* And likewise the transaction *f: service report* is decomposed into *f: delivery report* and *f: payment report*.

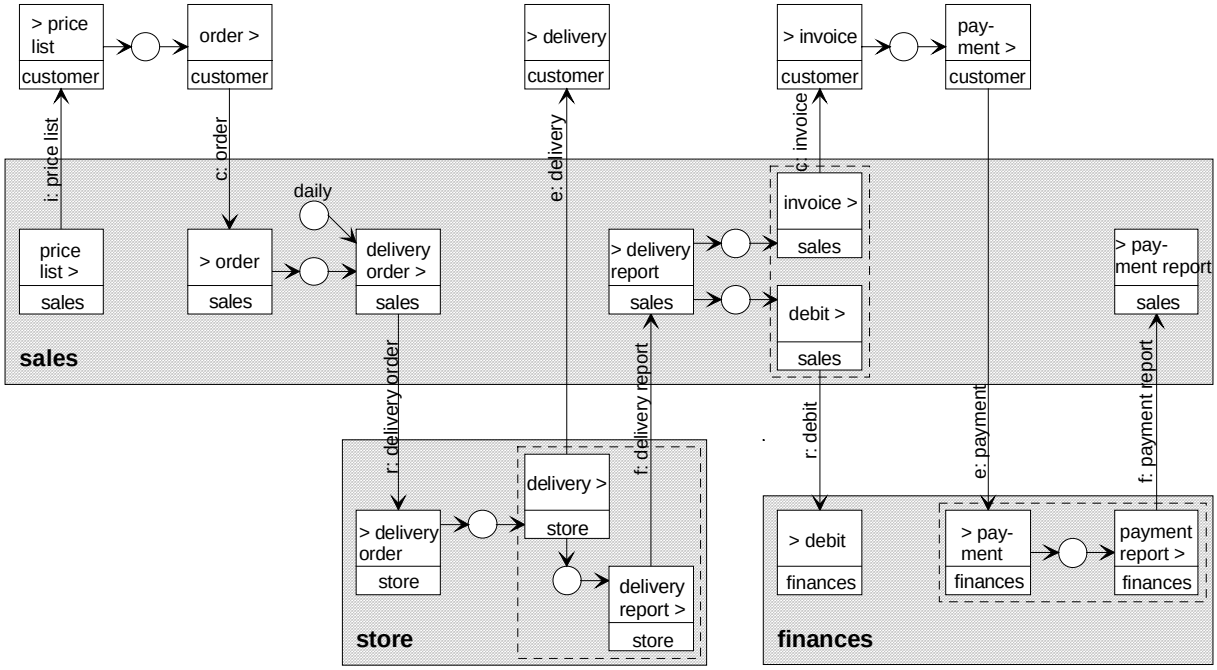

Fig. 10: Task-event schema of business process *distribution* (4th level)

5 Business application systems

5.1 Characteristics of Business Application Systems

As outlined in chapter 3, personnel and business application systems are resources to carry out business processes. Each task of a business process is carried out either by a person or by an application system or by both a person and an application system cooperating. This leads to the notion of **automation of tasks** [FeSi94]. A task

- is **automated (in full)**, if it is carried out completely by an application system,
- it is **not automated** if it is carried out by a person, and
- it is **automated in part** if it is carried out by both a person and an application system cooperating.

Similar considerations hold for the **automation of transactions** within information systems. A transaction

- is **automated** if it is performed by an electronic communication system and
- it is not automated if it is performed e.g. paper-based or orally.

Prior to defining the degree of automation, a task or a transaction have to be investigated if they are suitable for automation. A task is suitable for automation if its states and operations can be handled by a computer system. A transaction is suitable for automation if message passing and protocol treatment can be done by an electronic communication system.

The relationship between business process model and business application systems is exactly based on the concept of automation of tasks and transactions. The interaction schema of a business process model is convenient to record the extent of both the **achievable** and the **achieved degree of automation**. Figure 11 shows degrees of automation of tasks and transactions (see also [Kru97]) and applies them to the business object *sales* of the business process *distribution*.

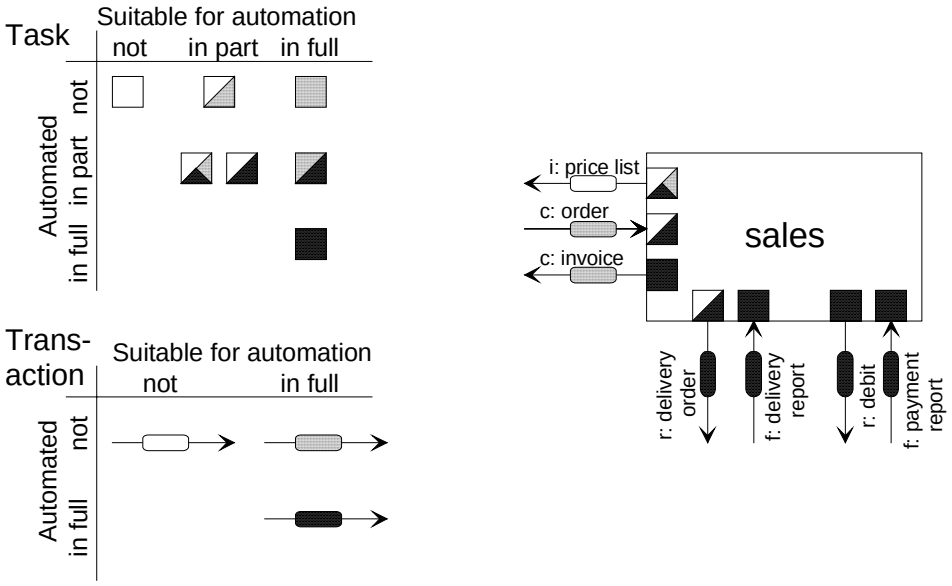

Fig. 11: Automation of tasks and transactions of the *sales* business object

The SOM methodology uses analogous structures of distributed systems at the business process model layer and the business application systems layer [FeSi96]. This simplifies the procedure of delimiting the set of tasks within a business process model which are to assign to a business application system. A network of tasks automated in full or in part finds a corresponding structure within application systems. The degree of integration of business application systems reflects the degree of connectivity within the network of tasks. There are no automation gaps and no functional redundancies.

Another effect of a balanced and synchronized development of business process model and business application systems is simultaneous evolution of both layers during their life cycle

[FeSi97]. There is a strong need that local changes in the business process model should only effect local changes in the business application systems. Both features, distributed systems at the two layers and the synchronized evolution, show that the business process model proves to be the backbone of a widespread architecture of business application systems.

5.2 Metaphor of Specifying Business Application Systems

To understand the way of specifying business application systems using the SOM methodology and to distinguish it from other approaches the underlying metaphor is pointed out. In short, the SOM methodology leads to (1) strictly object-oriented, (2) distributed, and (3) object-integrated business application systems [FeSi96].

Ad 1 Strictly object-oriented application systems: The object-oriented specification of business application systems fits the object-oriented characteristic of business process models. According to the SOM methodology the domain-specific specification of a business application system consists of a **schema of conceptual classes** and a **schema of task classes**.

Conceptual classes encapsulate (a) the states of the (automated) tasks of a business object as well as the states of the corresponding transactions and services, and (b) the operations defined directly and exclusively on these states. The initial structure of the schema of conceptual classes is derived from the most detailed level of the interaction schema in conjunction with the task-event schema of the corresponding business process model.

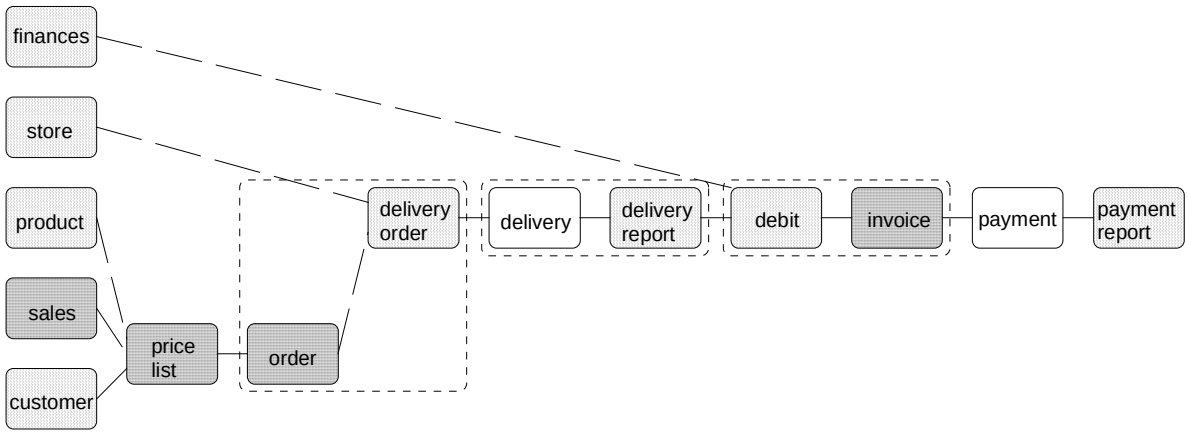

Fig. 12: Initial schema of conceptual classes of the business application system *sales*

Figure 12 shows the initial schema of conceptual classes derived from the business process model in figures 9 and 10. The classes at the left side correspond to the business objects and

the *product*. The class *price list* is derived from the corresponding transaction, connecting *sales* and *customer* with reference to *product*. The same way the other classes are derived from transactions. Figure 12 refers to the complete *distribution* process. The shaded classes belong to the *sales* application system. Dark shaded classes belong exclusively to the *sales* application system, light shaded classes are shared with other application systems.

Task classes specify the cooperation of conceptual classes and/or other task classes when executing a task automated in full or in part. In other words, task classes specify the work-flow within a business application system. The initial structure of the schema of task classes is almost identical to the most detailed level of the task-event schema of the corresponding business process model. Tasks lead to task classes, internal events and transactions lead to *interacts_with* relationships. Task classes derived from tasks which are automated in part must provide an interface for human-computer interaction. This allows a person to communicate with the application system when carrying out the task according to the partner-partner relationship.

Because of the structural analogy of task-event schema and schema of task classes, figure 10 illustrates the schema of task classes too. The shaded areas delimit the schema of task classes for the *sales* business application system as well as for *store* and *finances*.

Ad 2 Distributed business application systems: A distributed system is an integrated system which pursues a set of joint goals. It consists of multiple autonomous components which cooperate in pursuing the goals. There is no component which has global control of the system [Ens78].

In conjunction with the object-oriented characteristic of loosely coupled objects, each of them encapsulating states and operations, the SOM methodology leads to a specification of distributed business application systems in a very natural way. Initially, each conceptual class and each task class derived from a business process model is a candidate for an autonomous component. In this respect, the initial schema of conceptual classes and the schema of task classes show the maximum degree of distribution of a business application system relating to a given decomposition of a business process model.

During the specification process, the degree of distribution is likely to be decreased. This is done by merging two or more classes to one class. Classes may be merged due to domain-specific reasons like to reduce redundancy or to avoid sources of inconsistency (e.g. *invoice>*

and *debit*> in fig. 10, *debit* and *invoice* in fig. 12) as well as due to technical reasons to meet the capacity and performance of computer and communication systems.

Ad 3 Object-integrated business application systems: The way mostly used to integrate application systems is data integration. Several application systems share a common database, the functions of the application systems operate on this database via external views. Although this kind of integration preserves consistency and avoids redundancy of data, it is not sufficient to support flexibility and evolution of application systems.

This is why the SOM methodology completes the concept of data integration by the concept of object integration [Fers92, FeSi94]. This concept supports distributed application systems consisting of autonomous and loosely coupled sub-systems which themselves may be data integrated again. To achieve consistency of the application system as a whole, the sub-systems exchange messages according to detailed communication protocols. These protocols are derived from the transaction-oriented coordination of business objects as specified in the business process models.

5.3 Meta Model of Business Application Systems

The concepts used for the specification of object-oriented, distributed and object-integrated business application systems are now arranged to the meta model shown in figure 13. The notion of *class* follows the general understanding of object-orientation. Classes have *attributes* and *operators* and they are connected by binary *relationships*. Relationships are either *is_a*, *interacts_with*, or *is_part_of* relationships. *interacts_with* relationships denote channels for message passing between two classes, *is_a* relationships are used to model the specialization of a class using inheritance, and *is_part_of* relationships allow the specification of the component classes of a complex class.

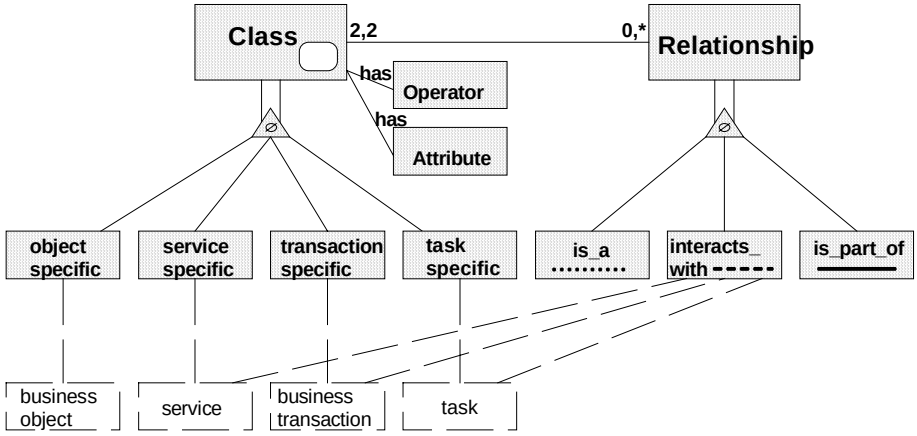

Fig. 13: Meta Model of Business Application Systems

In addition, figure 13 connects the meta model of business application systems to the meta model of business processes according to the architectural framework in figure 2. The relationships represented as dashed lines are used to derive the initial schema of conceptual classes and schema of task classes from a business process model. This way, a *class* is identified either as *object-specific*, *service-specific*, or *transaction-specific* when belonging to the schema of conceptual classes, or as *task-specific* (*task class* in short) when belonging to the schema of task classes. *is_a* relationships and *is_part_of* relationships cannot be derived from a business process model. They have to be included during the specification of the schema of conceptual classes or a schema of task classes, respectively.

6 Summary and outlook

The previous sections give a slim introduction to the SOM methodology for business systems modeling. A comprehensive enterprise model consists of sub-models for each layer of the enterprise architecture (fig. 1). The sub-models are balanced carefully according to the architectural framework (fig. 2). It is not necessary to start top down with the enterprise plan, followed by the business process model and ending with the specification of business application systems. Rather it depends on the goals pursued in the specific project where to start.

More and more, enterprise models prove to be indispensable for business engineering, information management, and organization. Enterprise models following the SOM methodology show several characteristics which support the management of large enterprise models: (a) several model layers, each focusing on specific characteristics of a business system, (b) definition of views on each model layer, outside and inside viewpoints, (c) different levels of ab-

straction and decomposition within a single model, and (d) notions with precise semantics which are arranged to meta models and based on appropriate metaphors. Compared to other approaches of enterprise-wide modeling, e.g. enterprise-wide data modeling, a comprehensive enterprise model offers big advantages and is more likely to be handled successfully.

There is a lot of research around the kernel of the SOM methodology which cannot be shown in this paper due to limitation of space. These feature include management of complexity (i.e. decomposition of large business process models into models of main and service processes), reuse of model components (using patterns, reference models, application objects), tool support (for modeling, reporting, business process management, information management, workflow management) [FeSi+94], an in depth consideration of distributed business processes and distributed business application systems [FeSi96] as well as first findings on virtual business processes [FeSi97].

7 Literature

Bahr92 **Bahrami H.:** The Emerging Flexible Organization: Perspectives from Silicon Valley. In: California Management Review, Summer 1992, p. 33 - 52

Beer81 **Beer S.:** The Brain of the Firm. 2nd Edition, Wiley, Chichester 1981

Ens78 **Enslow P.H.:** What is a 'Distributed' Data Processing System? In: IEEE Computer, Vol. 11, No. 1, January 1978, p. 13 - 21

Fers92 **Ferstl O.K.:** Integrationskonzepte betrieblicher Anwendungssysteme. Fachbericht Informatik 1/92. Universität Koblenz-Landau 1992

FeSi90 **Ferstl O.K., Sinz E.J.:** Objektmodellierung betrieblicher Informationssysteme im Semantischen Objektmodell (SOM). In: WIRTSCHAFTSINFORMATIK 32 (1990) 6, S. 566-581

FeSi91 **Ferstl O.K., Sinz E.J.:** Ein Vorgehensmodell zur Objektmodellierung betrieblicher Informationssysteme im Semantischen Objektmodell (SOM). In: WIRTSCHAFTSINFORMATIK 33 (1991) 6, S. 477-491

FeSi93 **Ferstl O.K., Sinz E.J.:** Geschäftsprozeßmodellierung. In: WIRTSCHAFTSINFORMATIK 35 (1993) 6, S. 589-592

FeSi94 **Ferstl O.K., Sinz E.J.:** Grundlagen der Wirtschaftsinformatik. Band 1, 2. Auflage, Oldenbourg, München 1994

FeSi95 **Ferstl O.K., Sinz E.J.:** Der Ansatz des Semantischen Objektmodells (SOM) zur Modellierung von Geschäftsprozessen. In: WIRTSCHAFTSINFORMATIK 37 (1995) 3, S. 209 - 220

FeSi+94 **Ferstl O.K., Sinz E.J., Amberg, M., Hagemann, U., Malischewski, C.:** Tool-Based Business Process Modeling Using the SOM Approach. In: Wolfinger B. (Hrsg.): Innovationen bei Rechen-

und Kommunikationssystemen. 24. GI-Jahrestagung im Rahmen des 13th World Computer Congress, IFIP Congress'94, Hamburg 28.8. - 2.9.94, Springer, Berlin 1994

FeSi96 **Ferstl O.K., Sinz E.J.:** Multi-Layered Development of Business Process Models and Distributed Business Application Systems - An Object-Oriented Approach. In: König W., Kurbel K., Mertens P., Preßmar D. (Hrsg.): Distributed Information Systems in Business. Springer, Berlin 1996, p. 159 - 179

FeSi97 **Ferstl O.K., Sinz E.J.:** Flexible Organizations Through Object-Oriented and Transaction-oriented Information Systems. In: Krallmann H. (Hrsg.): Wirtschaftsinformatik '97. Internationale Geschäftstätigkeit auf der Basis flexibler Organisationsstrukturen und leistungsfähiger Informationssysteme. Physica-Verlag, Heidelberg 1997, S. 393 - 411

Gamm+95 **Gamma E., Helm R., Johnson R., Vlissides J.:** Design Patterns. Elements of Reusable Object-Oriented Software. Addison-Wesley, Reading, Massachusetts 1995

Kru97 **Krumbiegel J.:** Integrale Gestaltung von Geschäftsprozessen und Anwendungssystemen in Dienstleistungsbetrieben. Deutscher Universitätsverlag, Wiesbaden 1997

Sinz96 **Sinz E.J.:** Ansätze zur fachlichen Modellierung betrieblicher Informationssysteme. Entwicklung, aktueller Stand und Trends. In: Heilmann H., Heinrich L.J., Roithmayr F. (Hrsg.): Information Engineering. Oldenbourg, München 1996, 123 - 143

Sinz97 **Sinz E.J.**: Architektur betrieblicher Informationssysteme. Erscheint in: Rechenberg P., Pomberger G. (Hrsg.): Handbuch der Informatik, Hanser-Verlag, München 1997

VoBe96 **Vossen G., Becker J. (Hrsg.):** Geschäftsprozeßmodellierung und Workflow-Management. International Thomson Publishing, Bonn 1996