144

P. Loos, Th. Allweyer

Process Orientation and Object-Orientation — An Approach for Integrating UML and Event-Driven Process Chains (EPC)

March 1998

Process Orientation and Object-Orientation —
An Approach for Integrating UML and Event-Driven Process Chains (EPC)

Peter Loos , Thomas Allweyer
(loos@acm.org, t.allweyer@ids-scheer.de)

March 1998

Paper 144
Publication of the Institut für Wirtschaftsinformatik
University of Saarland, Saarbrücken, Germany
http://www.iwi.uni-sb.de/public

(98-04-02)

Contents

Abstract

Event-driven process chains (EPC) are used by many companies for modelling, analysing and redesigning business processes. The resulting EPC models are used as a starting point for the development of information systems and for the definition of workflows. They can be applied for simulation and activity based costing. A major area of application is the implementation of standard software packages, because many vendors have documented their software's processes with EPCs. Since the unified modeling language (UML) as the new standard for modelling object-oriented systems does not yet cover all aspects required for the above mentioned application areas, an integrated approach is required for describing the dependencies between event-driven process chains and UML diagrams. In this paper, such an integrated approach is presented. It focuses mainly on the connection between EPC, class diagrams, and statechart diagrams, but the relationships between EPC, use case diagrams, and activity diagrams are also discussed.

1 Introduction

The successful development and implementation of business information systems requires an integrated approach which includes the seamless design of both the business processes and the information systems supporting the business processes. Therefore, several frameworks and modelling methods have been developed for an integrated modelling of the entire enterprise with respect to both organisational and information systems aspects (cf. Olle et al. 1988, ESPRIT 1989, Martin 1989, Österle 1995, Scheer 1998a). Due to the architecture of most existing business information systems, these approaches were usually based on traditional software development paradigms rather than on object-orientation. Object-oriented modelling methods, on the other hand, used to cover only aspects which are close to implementation, but not the business processes. Currently, however, these two worlds are moving closer together. There are several reasons for this:

- Object-oriented implementation languages, such as C++ or Java, play an increasingly important role for the development of business information systems. In order to give an overall description of a company's business processes and its object-oriented information systems it is therefore necessary to integrate business process modelling languages with object-oriented languages.

- Object-orientation becomes more and more popular not only as a software paradigm, but also as a way of mentally structuring complex systems. Thus, object-oriented approaches can also be used on the business level for identifying the objects to be dealt with in a business process (such as customer, product, order) and the objects' relationships (cf. Jacobson et al. 1994, Taylor 1995).

- With the development of business objects as software components which can be assembled to an information systems according to the user's specific requirements, it is necessary to provide means for defining how a business process is supported by the cooperation of different business objects (cf. Spurr et al. 1994).

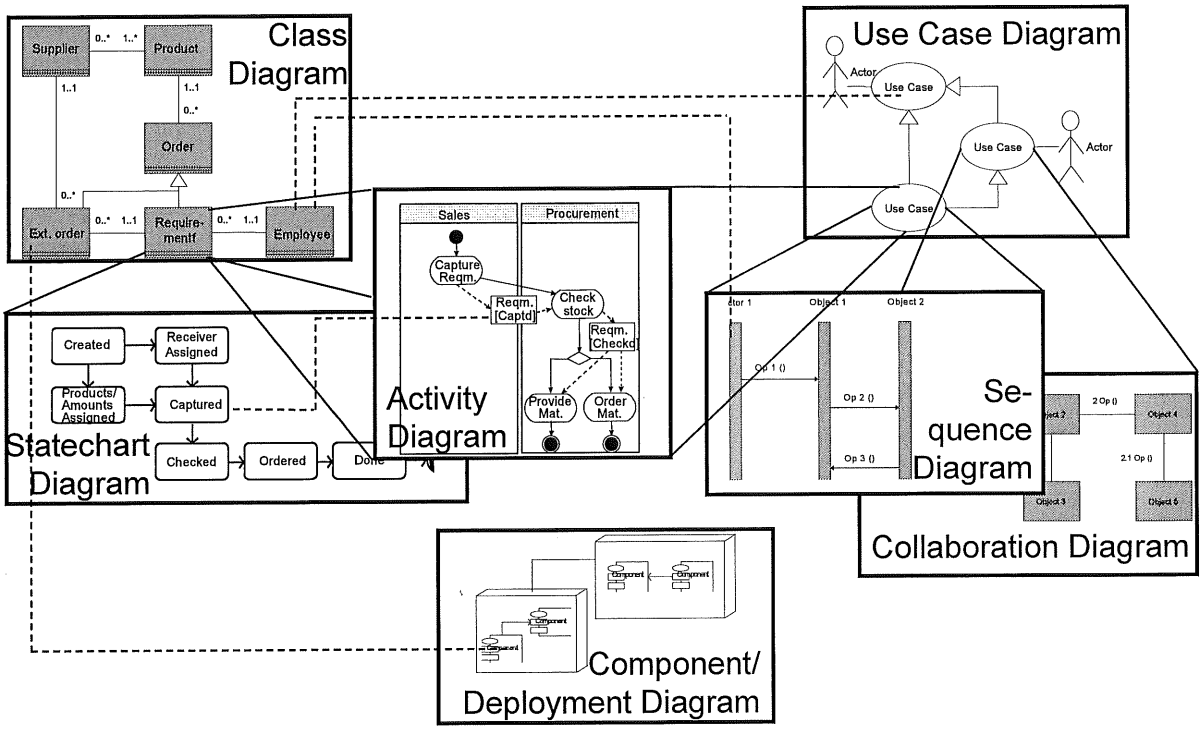

Fig. 1: Diagrams of the Unified Modeling Language (UML) and their connections

The Unified Modelling Language (UML) as the new standard for object-oriented modelling combines the most important existing object-oriented approaches. Fig. 1 shows the types of diagrams provided by the UML and their main connections. For a description of the UML, see Rational 1997b, 1997c, Fowler/Scott 1997.

The developers of the Unified Modeling Language (UML) have recognised the need for modelling methods which cover the end-users' views and which allow for modelling processes. Therefore diagrams like the use case diagram and the activity diagram have found their way into the UML (cf. Rational 1997b and 1997c), as well as some specific stereotypes for business modelling (cf. Rational 1997a). However, not all concepts which are relevant for modelling, analysing and designing business processes are covered by the current version 1.1 of UML (cf. Amber 1997). It is thus necessary to combine the UML with powerful state-of-the-art business process modelling languages (cf. Scheer et al. 1997, Nüttgens et al. 1998, Ovum 1997 and 1998).

Such a business process modelling language is the event-driven process chain (EPC). The EPC provides comprehensive means for modelling different aspects of a business process. It is mainly used for:

- business process re-engineering (BPR)
- definition and control of workflows
- configuration of standard software
- software development
- simulation
- activity based costing (ABC)
- quality-related documentation of processes according to the requirements of ISO 900x

There are several thousand companies worldwide who have modelled their business processes with EPCs. Several providers of standard software packages (among them SAP) have used the EPC for documenting the business processes which can be supported by their software. The end user companies can adapt these software models to their specific requirements in order to customize the system.

This paper describes an approach for integrating the EPC with the UML. With this approach it is possible to model all relevant aspects of a company's business processes and its object-oriented information systems without the need for switching between different modelling paradigms or for translating between different modelling languages.

In the second paragraph, the notation of EPCs and their application to business process modelling are described. An overview over existing approaches for integrating the EPC with object-oriented methods is also given. Paragraph three discusses the usability of different kinds of UML diagrams for business process modelling, and it identifies useful modelling concepts which are not yet covered by the UML. The integration of EPCs with class diagrams and statechart diagrams is presented in paragraphs four and five. Paragraph six discusses the relationships between EPCs and other UML diagrams, namely use case diagrams, activity diagrams, sequence and collaboration diagrams.

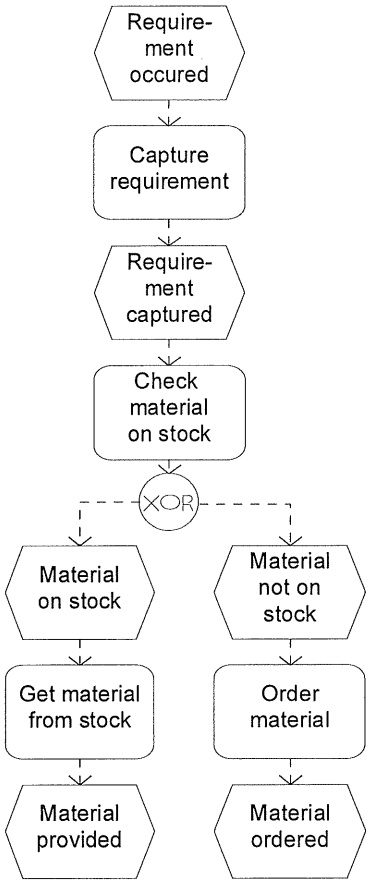

Fig. 2: Event-driven process chain (EPC)

2 Modelling Business Processes with Event-Driven Process Chains

The main elements of an event-driven process chain (EPCs) are functions and events. Functions are triggered by events, and functions produce events. The control flow of a business process is therefore described by a sequence of alternating events and functions (cf. Keller et al. 1992, Scheer 1998b). Alternative or parallel paths can be modelled with logical operators, such as AND, OR, XOR, or more complex expressions. These operators can be used for splitting and joining the control flow. An example of an EPC is shown in Fig. 2.

2.1 EPCs in an Architectural Framework

Although the control flow is the most important aspect for describing a business process, there are many other kinds of information which can be relevant, depending on the modelling purpose (e.g. BPR or workflow definition). Such kinds of information include the people and organisational units responsible for carrying out a certain function, the (material or immaterial) output-product, as well as the data flow. In the centre of Fig. 3, the EPC from Fig. 2 has been extended with data, products, and organisational elements (therefore sometimes called extended event-driven process chains - eEPC). For each function it is shown which organisational role carries out this function. These roles are defined in an organisational chart of the company which is shown in the upper part of Fig. 3.

The data flow is shown with in- and output connections between the EPC's functions and entity types and relationships from a data model. This data model is shown as an entity-relationship model (ERM) on the left. In-

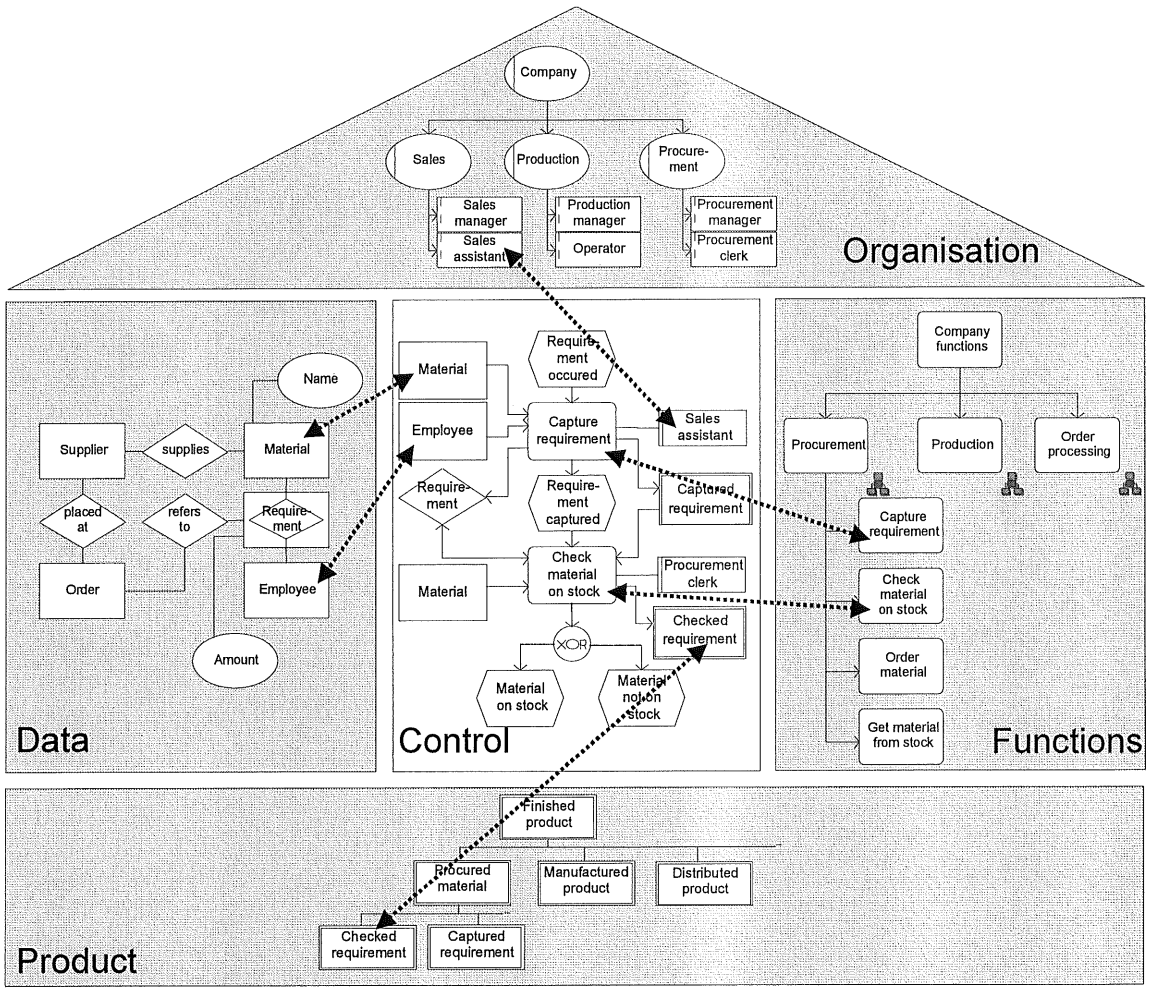

Fig. 3: Extended EPC in ARIS

stead of using entity types, it is also possible to define the data flow on a more detailed level between functions and attributes or - on a more aggregated level - between functions and data clusters. The EPC functions can also be included in a hierarchical function tree, as it is shown on the right of Fig. 3.

The objective of a business process is the creation of a product. A product can be a material item, like a machine, or an immaterial product like a service. Besides the output aspect, products are also relevant as inputs for business processes. The mutual relationships between products can be depicted in hierarchical and non-hierarchical diagrams like in gozintographs which are used for the graphical representation of bills of materials in the manufacturing sector. The structure of products is shown in the bottom part of Fig. 3.

The different types of diagrams are all mutually related. They provide different views on the overall system: the organisational view, the functions view, the product view, the data view, and - integrating the other views - the control view. This structure of integrated views is part of the "Architecture of Integrated Information Systems" (ARIS, cf. Scheer 1998a).

EPCs can be hierarchically structured across any number of levels by assigning more detailed EPCs to every function within an EPC. Such a detailed EPC denotes the process which is carried out when the respective function is triggered. Functions for which it is not feasible to define an even more detailed process, are called "elementary functions".

Events can also be described with so-called event-diagrams, in which a complex event (such as "order checked") is decomposed into detailed events (such as "customer data checked" and "product data checked"). An event can be connected to data view elements, such as attributes, so that it is possible to model the fact that an event occurs when both the preceding function is finished and a specific attribute has a certain value. An event can also be directly related to a product, if the event just describes the occurrence of a function's output.

In addition to functions, events, data objects, products, and organisational elements, a wide range of information can be included into an EPC in order to describe specific aspects more clearly. Such types of information include:

- The use of different kinds of information carriers and communication media, such as paper documents, floppy disks, telephone etc.
- Information systems used for supporting functions
- Human know-how and qualifications
- In- and output material of a function (e.g. in the production)
- Required equipment or machinery (resources)

In many cases, there can be different types of connections between two kinds of elements. Between an organisational unit and a function, for example, there can be several connection types, such as:

- Organisational unit *carries out* function
- Organisational unit *is responsible for* function
- Organisational unit *provides support for* function
- Organisational unit *must be informed about* function
- Organisational unit *must approve of* function

These different kinds of connections can be used for modelling in detail how different organisational units work together in order to perform the functions of a process.

Only a part of the information about a business process is shown in the EPC diagram, because each element and each connector possesses various characteristics which are expressed through attributes. There are general at-

tributes which are common to all types of elements, such as the identifier or the description, but most of the attributes are type-specific, e. g. for a function there are attributes concerning the time (such as the minimum, the maximum and the average times for processing, waiting, and transfer) and costs. Connectors can also have attributes, such as probabilities for control flow connectors.

Not all of the rich semantics can and should be used within a single project. The types of elements, connectors and attributes to be used depend on the modelling purpose. In an EPC which should be simulated it is necessary to define probabilities and resource capacities, while for a workflow EPC it is important to assign people and roles to the functions. In most projects in which EPCs are used, the appropriate set of modelling constructs is defined in modelling guidelines.

2.2 Object-Oriented Business Process Modelling Approaches

Since object-orientation becomes more and more popular to non-software experts, there are several approaches which combine object-orientation with a business oriented view (e.g. Ferstl/Sinz 1996). There are two approaches specifically dealing with the EPC and object-orientation.

The first approach utilises the object-oriented paradigm to achieve a more powerful description of business processes (Bungert/Heß 1995). This is done by employing the

- object-oriented class concept and data encapsulation, the
- message concept, and
- object hierarchies and inheritance,

to extend the metamodel of the EPC. The class concept and data encapsulation are realised by complementing the data items (entities or data clusters) with functions from the EPC in separate class diagrams. EPCs' events are also incorporated in the class diagrams as function outputs and as functions' pre-conditions. The message concept is introduced by considering how functions are connected with each other via events. Identical events within different object classes indicate message based object interaction, e.g. if the event E1 is produced by function F1 within the class diagram of class C1 and event E1 triggers function F2 within the class diagram of class C2, this leads to the conclusion that there is a message based interaction between C1 and C2 respectively between F1 and F2. For object hierarchies and inheritance, the generalisation and specialisation constructs of the entity-relationship-model are used. Several steps are recommended to apply the approach:

1. define the object classes that are relevant for the business processes
2. assign the processes' functions to the object classes
3. refine the class structure based on the functions that could not be assigned in the preceding step
4. define start events for each function
5. define for each event the functions it is produced by
6. use the functions and events to derive EPCs
7. if the level of detail is not sufficient, iterate from the second step

The second approach modifies the EPC diagram itself (Scheer et al. 1997). The purpose is to show both the event-driven process chain and the corresponding interaction of business objects in the same diagram. As usual, business object classes own instance attributes and methods (respectively functions). For modelling the business processes, the object interaction via message exchanges is denoted as event-driven control-flow. Two different types of messages for control flow are distinguished:

- Event-based messages describe the information about state transitions. They are explicitly modelled by events. Hence object classes are connected via events in order to describe a business process.
- Service-control messages are used to define client-server-relationships among classes. The sender (client) triggers the recipient (server) to deliver a service. Usually there is no event modelled between a client object

and a server object since the server object is regarded to be involved only transitively in a business process. However, the explicit modelling of events for this type of relationships is possible.

The diagram for the object-oriented business process, called object-oriented event-driven process chain (oEPC), does not look very different from the common EPC, but functions are replaced with object classes which include the business functions. It is also possible to assign object methods to the object classes within the oEPC to indicate what functions have to be carried out by the object at that particular business process step. Additional to that, further aspects can be included like in the extended EPC, e.g. organisational responsibility.

While the first approach does not change the EPC, but it introduces own class diagrams with EPC elements, and the second approach introduces a further development of EPC, the approach presented in the following paragraphs aims to address the following integration issues:

- It is focused on the relationship between existing EPC elements and UML elements rather than on the definition of a further method.
- No strong formal constraints should restrict the user in the requirements definition phase. The integration should leave the same openness to the usability as the common EPC.
- The existing structures of EPCs should be preserved so that a mainly process oriented employment of the EPC is still supported. This will also guarantee a smooth transition between the process orientation of the EPC and object-orientation.
- The integration with the EPC should be achieved without modifying the standard UML definitions.

3 Usability of UML Diagrams for Modelling Business Processes

When discussing the relationship between business process modelling and object-oriented modelling it is necessary to clarify the purpose of the modelling activity in respect to the scope of the universe of discourse.
Fig. 4 illustrates the range of a business process compared to the scope of an object system.

- The smallest granularity of a process description is object internal, as shown on the left side of Fig. 4. Such processes might be handled completely by one object operation or by several operations of one specific object. To model this aspect of process descriptions, UML employs state machine techniques by providing statechart diagrams and activity diagrams. Since these diagrams describe the behaviour of the system, they cover at least some aspects of a process description.

- The next step of granularity ranges beyond a single object, but stays inside the object system that implements the application software. Therefore this granularity is called object system wide. From the perspective of business process modelling all process functions are implemented through software code respectively through object operations. In UML, interaction diagrams show patterns of interaction among objects. There are two forms of interaction diagrams, the sequence diagrams and the collaboration diagrams. While sequence diagrams emphasise the interaction in time sequence, collaboration diagrams focus on the object roles of the relationships. Both are suitable to cover some aspects of object system wide business processes.

- The largest granularity occurs when business processes range beyond the scope of an object system. An example is depicted on the right side of Fig. 4. Some parts of these business processes may be carried out by other object systems, but most of these activities are done manually without any or with only little system support. In UML, use case diagrams could be used to model aspects which are beyond the object systems, since use cases describe relationships with external interactors, and they are also used to depict a complete enterprise (cf. Jacobson et al. 1993).

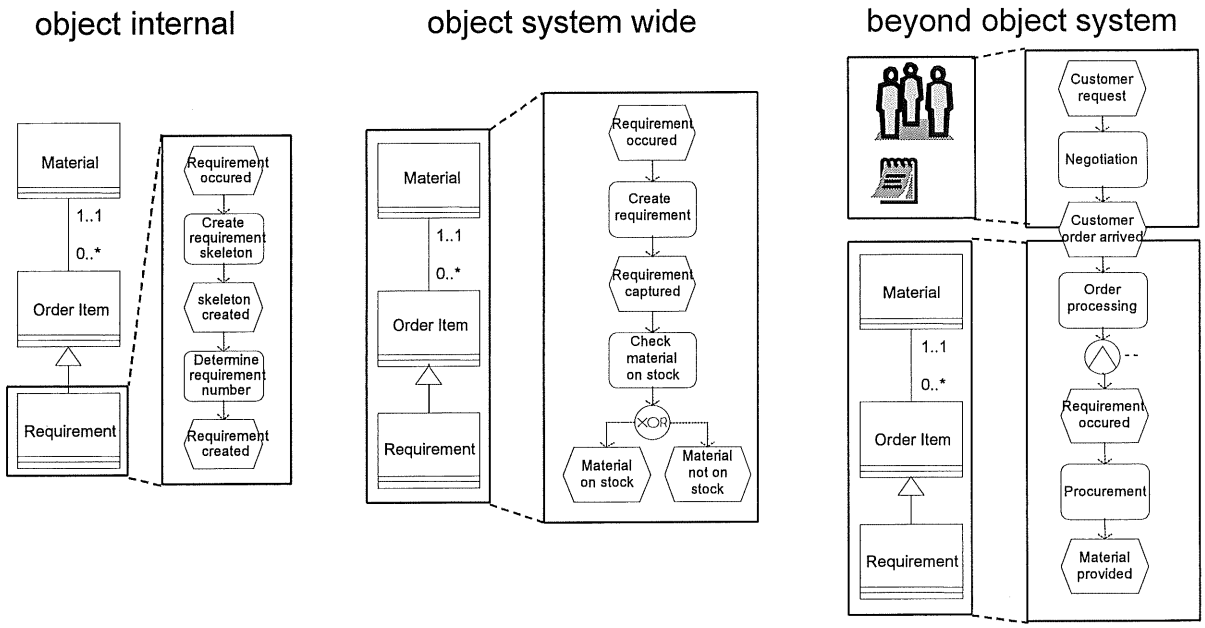

Fig. 4: Granularity of process description compared to scope of object system

This classification according to granularity gives an indication for applying UML behavioural diagrams for modelling business processes. However, some more investigations have to be done to analyse whether UML diagrams are sufficient to cover all aspects of business processes as established methods do.

Use cases diagrams cover some organisational aspects, especially the interaction with organisational units external to the object system. However, use case diagrams do not offer mechanisms for depicting the process flow, they particularly do not provide means for defining the sequence of activities and conditions for the activation of process activities. The internal flow of use cases has to be described in textual form.

Sequence diagrams and collaboration diagrams offer some aspects of business process flow, but they focus on the interaction between objects and not on the depiction of an overall control flow. Furthermore, these diagrams are very close to implementation. They are useful for the modelling of technical details. While it is possible for end users to get familiar with more straightforward business modelling methods, the detailed description of message exchange in sequence diagrams and collaboration diagrams is not suitable for non-software experts. Finally, the diagrams do not offer a real support for organisational purposes.

Statechart diagrams are usually assigned to one object. This scope is too narrow for most business processes unless it can be assured during design that single object classes match exactly the respective business processes. For business processes with a higher granularity it is not feasible to define all conceivable states (in the sense of state machines), a business organisation can take on. Besides that, no organisational aspects can be modelled.

At the first glance, activity diagrams seem to be very close to business modelling methods like the EPC, since they offer components like activities, relationships for control flow, logical connectors for control flow variations. The elements can be grouped in swimlanes in order to assign organisational units. Compared with EPCs, there are some problems connected with the use of activity diagrams for business process modelling:

- Not all logical operators for splitting and joining the control flow can be modelled in a straightforward way. There are components for AND and XOR, but no equivalent for the inclusive OR connection.
- The organisational responsibility for activities can be expressed by placing the activities in swimlanes. However, swimlanes are not sufficient for modelling advanced and precise organisational relationships as men-

tioned before. They are important e.g. for the definition of workflows when support through workflow management systems is intended and for ISO900x-related quality documentation.

- Although there is no inherent reason, UML does not consider activity diagrams for modelling object external flows.
- There is no support for modelling additional information like it is available in the EPC, e.g. information carriers or required materials and other resources.
- The definition of activity diagrams as state machines is problematic for applying activity diagrams according to the UML definition for business process modelling, since not all business functions in a company can be regarded as object internal action states.

Since UML behavioural diagrams do not cover all aspects required for business process modelling, the integration of business process modelling techniques is necessary. As suggested in Fig. 4, the EPC can be applied on all levels of granularity for business process modelling. Therefore, the relation of the EPC and UML diagrams should be explored, since the UML - if enriched with the EPC - could provide useful support for integrated modelling of business processes and object-oriented information systems.

4 Connecting Class Diagrams with EPCs

The most important connection of the EPC is the one to the class diagram, since class diagrams are the central part of the UML and the foundation for the actual implementation. By connecting EPCs and class diagrams it is possible to define which kinds of objects are required, created or changed by a function, and what operations and attributes are needed. Thus it is possible to identify relevant classes based on an EPC. If there is already an existing class diagram (e.g. of the business objects provided by a software package), the EPC can be used to define how these existing classes are used for supporting a specific business process.

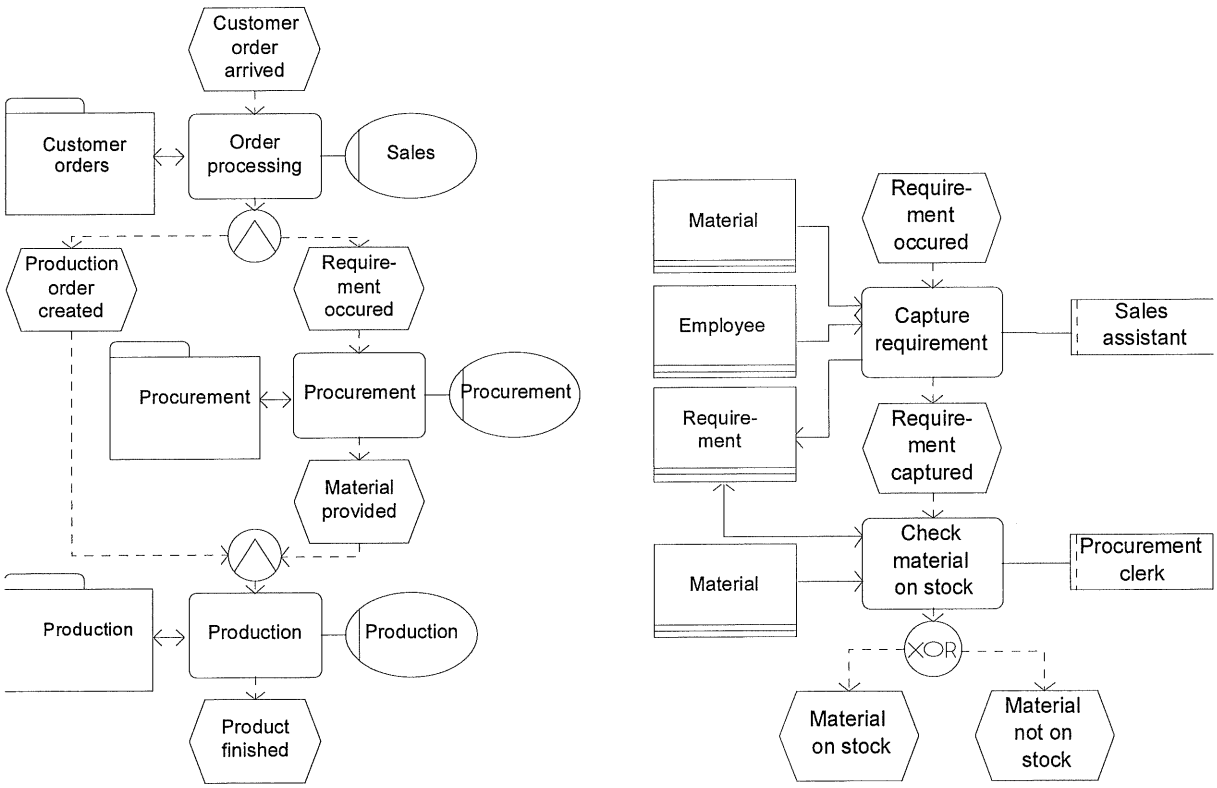

Fig. 5: High-level EPC with packages (left) and medium-level EPC with classes (right)

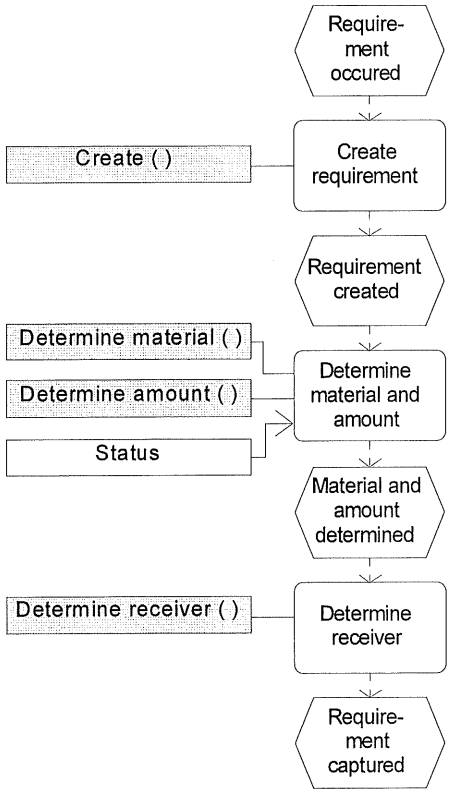

Fig. 6: Detailed EPC "capture requirement" with operations and attributes

On a high level, in- and output-connections between functions and packages can be defined (cf. Fig. 5, left side). An input-connection, denoted by an arrow from a package to a function defines that the function requires information stored with objects of the package's classes. These objects are not changed by the function, if there is not also an output-connection from the function to the package.

According to the principles of object-orientation every interaction with an object is done via messages, and it is not possible to determine whether a certain message actually leads to a change of the object or not. Therefore, the distinction between input- and output-connections does not define a hard requirement for the implementation, but it expresses the intention of how the respective objects are primarily used by the function, i.e. if the function only needs to read information which is stored with a certain object, or if the purpose is to create or modify the object, or both.

On a more detailed level, functions are connected directly with classes rather than packages, in order to express that a function uses or modifies objects of a certain class. Fig. 5 shows an EPC on the right side which details the function "procurement" from the EPC on the left side. Accordingly, the classes used in the EPC on the right side are contained in the package "procurement" on the left side.

The functions from the EPC of the right side of Fig. 5 can be further detailed, as it is shown for the function "capture requirement" in Fig. 6. On this level, it can be useful to define the operations and attributes to be used. If an operation is connected with a function, this means that the operation is called during the execution of the function. It is also possible to define in- and output-connections between functions and attributes. In the actual implemented OO-system it is of course not possible to access attributes directly, but only via appropriate operations. However, for the development of business processes and the main class structures, it can be rather convenient just to link an attribute to a function, so that it is not necessary to model standard "get-" and "set-"operations explicitly. During the business process design and the basic structuring of a business-related class diagram, modelling such details would rather distract from the essential contents of the model. These details

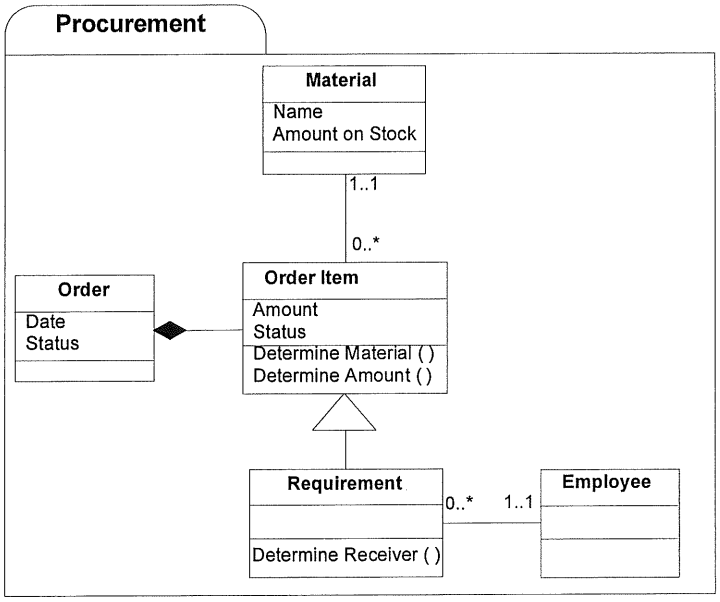

Fig. 7: Class diagram

should be added later on in a more implementation-oriented class diagram. Such an implementation-oriented class diagram is developed from the business-oriented diagram by adding technical classes and operations.

In a detailed EPC, functions are usually very close to single operations. If a function consists only of the call of one operation, it is not necessary any more to distinguish between the function and the operation, but the operation itself can be used as a function. In Fig. 6, the function "create requirement" does nothing more than calling the operation "create" from the class "requirement" (The operations used in the EPC are connected to classes in the class diagram). Instead of modelling a function connected with an operation it would therefore be possible to use directly the operation instead of the function, and connect it to the triggering and resulting events. In general, a business function of an EPC can be equivalent to an operation of a class. If S is a set, and S(function) is the set of functions, then:

$S(function) \cap S(operation) \supseteq \varnothing$

However, the equivalence is not mandatory. There can also be functions with no equivalent operations:

$S(function) \setminus S(operation) \supseteq \varnothing$

This occurs for business functions on a high aggregated level or functions beyond the scope of the object system (but it should be noted that the object system does not need to be limited to the scope of the information system). There are also operations with no equivalent functions:

$S(operation) \setminus S(function) \supseteq \varnothing$

This is especially true for operations which focus on a very detailed level or which cover technical and implementation items.

Ideally, each operation which is relevant from a business perspective should also occur as a function in an EPC. Thus, the gap between functions and operations on the requirement definitions level will be very small.

The packages, classes and operations used in the above EPCs are all defined in a class diagram, as it is shown in Fig. 7.

EPCs can also be used to depict the process within an operation. In many cases, an operation may comprise a sequence of activities which is a relevant process from a business process perspective. For example, a class "production order" may have an operation "release". When this operation is called, the completeness and consistency of the production order is checked, the availability of material and required equipment is tested, etc.

Since this process is not only relevant for the software engineer, but also for the production manager, it should be documented as an EPC.

The hierarchy of refinement of the EPCs shown in Fig. 5 through Fig. 6 with its examples of object-EPC-relationships may correlate with the granularity of business processes of Fig. 4, but this is not mandatory.

5 Connecting Statechart Diagrams with EPCs

UML statechart diagrams can be used for modelling an object's life-cycle by defining its relevant states and possible state transitions. Such a statechart diagram is usually not drawn for every class, but only for those classes which require a detailed analysis of their states and transitions, e.g. in rather complicated cases.

The EPC of the right side of Fig. 5 contains in- and output-connections between functions and classes, in order to describe that a function uses or modifies objects of these classes. However, it may not be sufficient for the function to have any object of a specified class, but it may be also important that the object is in a certain state. For example, the function "order material" requires that the input-object of the class "requirement" has been checked before, i.e. it must be in the state "checked". The function "check material on stock" transfers a requirement from the status "captured" into the desired status "checked".
The information about objects' states can also be modelled in an EPC by drawing in- and output-connections between functions and object states rather than between functions and classes. The relationship between an EPC and a statechart diagram is shown in Fig. 8. The right side contains the statechart diagram of the class "requirement". On the left side, the use of the object in its different state is shown. The rectangular object states in the EPC refer to the states in the statechart diagram. Each EPC function which transfers the object from one state to another carries out one of the transitions defined in the statechart diagram. EPC and statechart diagram must be consistent, i.e. a function cannot transfer an object from one state into another, if this is not possible according to the transitions defined in the statechart diagram.

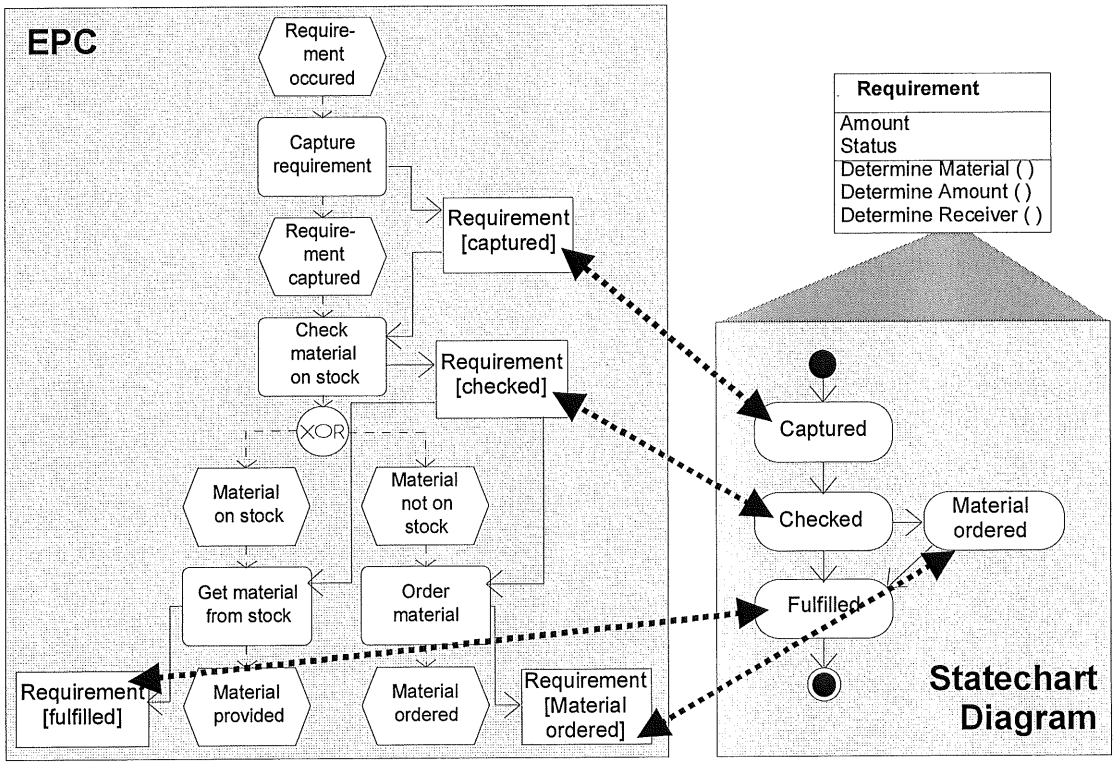

Fig. 8: Use of states in an EPC

It should be noted that there is some redundancy between the use of states and events in an EPC. In many cases, an event denotes that an object has changed to a new state. For example in Fig. 8 the event "requirement captured" is redundant to the output-object "requirement" in its state "captured". In such cases it is sufficient to use only one of both constructs. In other cases, events and states are different, such as the requirement's state "checked" with the two related events "material on stock" and "material not on stock".
Object states in an EPC can be used to define products. In general:

$$S(\text{product}) \cap S(\text{state}) \supseteq \varnothing$$

This holds especially true for non-material products, where the output product of a function is an object in a certain state. Therefore, there is no benefit of using both products and states in an EPC simultaneously. Rather it is recommended to regard the product and the state (respectively the object in a certain state) as identical, e.g. the object state "requirement [captured]" of Fig. 8 is indeed the same as the product "captured requirement" of Fig. 3.

6 Relationships between EPCs and other UML Diagrams

Each of the UML diagram types discussed in paragraph 3 (use case diagram, sequence/collaboration diagram, activity diagram) contains some aspects which are also relevant for business process modelling, i.e. there is some overlap between these diagram types and the EPC. However, since each diagram type has a certain focus, it can be useful to work with several or all of these diagrams in order to make clear different aspects of the business processes and the underlying information system. Therefore it is necessary to define how the EPC relates to each of these diagrams.

There can be two kinds of relationships between the EPC and the process-related UML diagrams:

1. The first kind of relationship is relevant if both diagrams are used together, each for describing a different aspect. Such a relationship can include the use of references to the same kinds of elements (e.g. the same operation may be referred to in two diagrams of a different type), and the detailed description of a model element by a diagram of another type. These kinds of relationships exist also between different UML diagrams.

2. The second kind of relationship is the definition of translation rules from one notation to the other. This can be useful for converting process descriptions familiar to business process experts to models which are familiar to software developers. In many cases it also necessary to make such a translation in order to transfer a model from a business process modelling tool to a CASE tool, or vice versa.

6.1 Use Case Diagrams and EPCs

Between use case diagrams and EPCs, the first kind of relationship is the most relevant. Although it would be possible to convert EPC functions, roles and their relationships to use cases, actors and communicates-relationships, a lot of the EPC semantics would get lost, since use case diagrams do not contain any control flow.

Use cases usually do not represent business-related functions, but interactions with a software system. For carrying out an EPC function, such as writing a letter, there may be several use cases, e.g. editing, formatting and printing text. Therefore, it is possible to detail an EPC function with a use case diagram (cf. Fig. 9, left).

On the other hand, it is also possible to define rather comprehensive use cases, such as "customer order processing". The underlying sequence of activities, i.e. the actual business process, is often only described verbally (cf. Jacobson et al. 1993 and 1994). For these cases it is useful to assign an EPC to a use case in order to provide a model of the underlying process rather than just a verbal description (cf. Fig. 9, right).

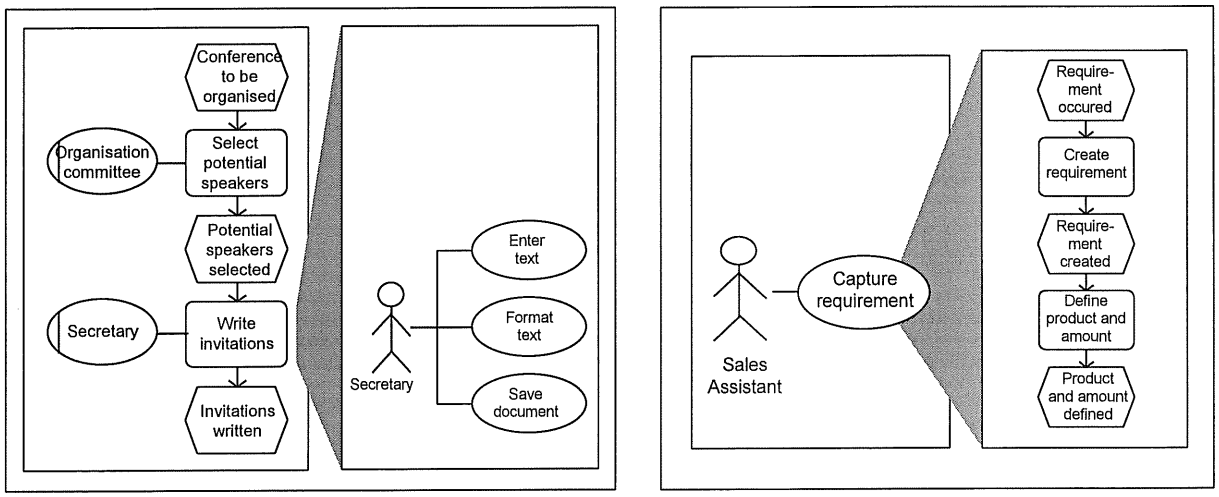

Fig. 9: Detailing of functions with use case diagrams (left) and detailing of use cases with EPCs (right)

In both cases - detailing a use case with an EPC and detailing a function with a use case diagram - it is possible to use the same roles (respectively actors) in both diagrams. These roles can be defined in the organisational chart.

6.2 Sequence/Collaboration Diagrams and EPCs

Sequence and collaboration diagrams are much closer to implementation than EPCs. While EPCs may contain rather loosely defined business functions, and they include only those object types and functions that are relevant from a business perspective, sequence and collaboration diagrams describe the detailed message exchange in a much more formalised way. It is usually not possible to directly translate EPCs into sequence/collaboration diagrams although there is some redundant information in both diagrams.

It is therefore the responsibility of the modeller to ensure that the message exchanges depicted in a sequence or collaboration diagram are consistent with the business processes to be supported. EPCs and sequence/collaboration diagrams are mainly connected via the use of the same elements, i.e. object types and op-

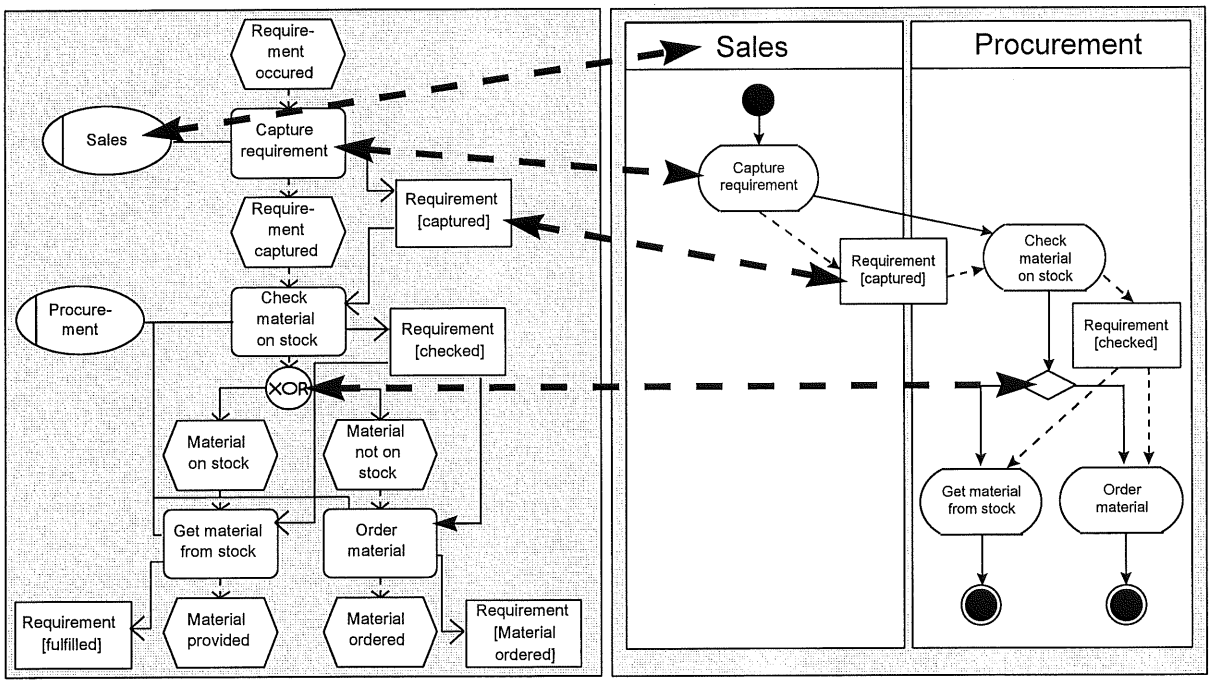

Fig. 10: Translating between EPCs (left) and activity diagrams (right)

erations.

6.3 Activity Diagrams and EPCs

Since activity diagrams and EPCs look rather similar it is usually not advisable to use both diagram types together to cover the same part of the universe of discourse (although this would be possible, and diagrams of both types could contain references to the same elements). However, it is possible to translate activity diagrams into EPCs and vice versa (cf. Fig. 10). Of course, some types of information in an EPC (such as material flow or information carriers) get lost in such a translation, since activity diagrams do not allow for modelling these features.

7 Procedural Models for Applying the Integration

The advantage of the integration of UML and EPC is that there is a well-defined transition between a process oriented analysis step and object-oriented design and implementation steps. Furthermore, the process oriented view can also be used in the design step. Fig. 11 illustrates two procedural models for applying the proposed integration.

- The procedual model for business process modelling projects is depicted on the left. In a first step, value-added chains with the core functionality to meet the business objectives are defined. Based on the value-added chains, first processes can be developed by applying the EPC method. The high-level EPCs are usually developed in a top-down approach, but a mixture with a bottom-up approach can also be suitable. As mentioned before, class diagrams have the most important connections to the EPCs. Therefore, detailed EPCs and class diagrams are developed together in an iterative process based on the input from the high-level EPC. Statechart diagrams may also be developed if it is desirable to express explicitly the states objects can be in. It should be stressed that the class diagrams should focus on a business oriented view to the universe of discourse, not on technical or implementation aspects, i.e. these class diagram contain business objects respec-

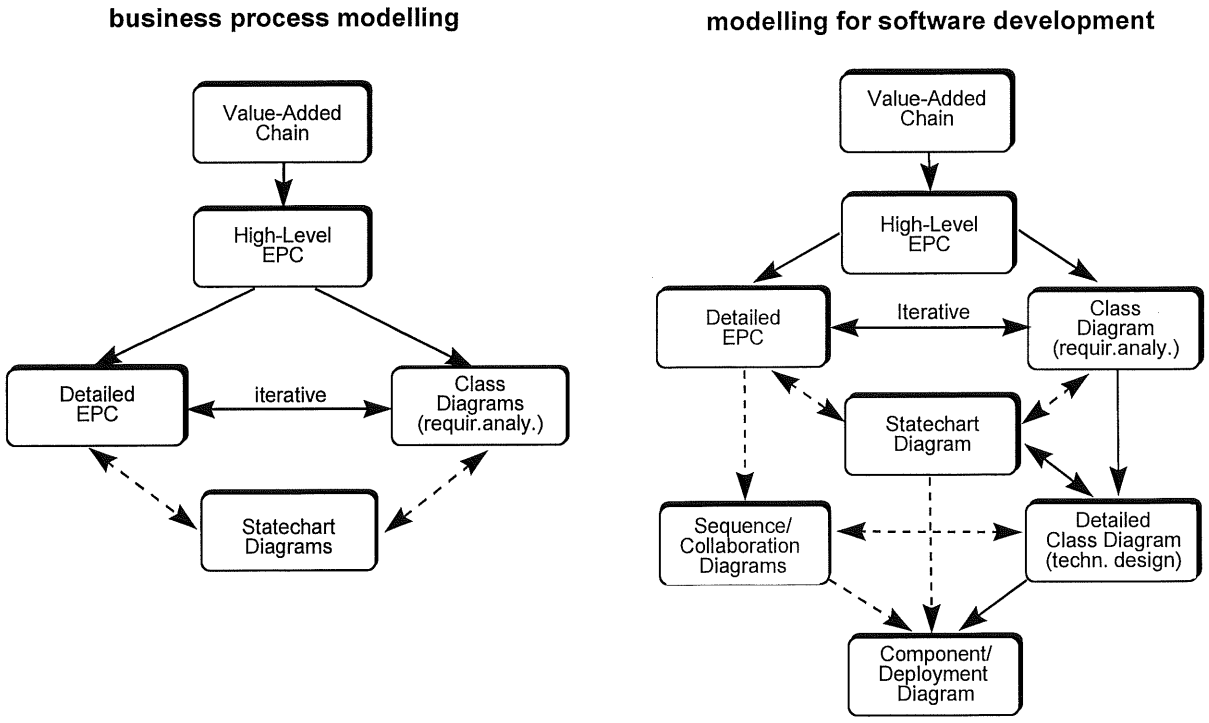

Fig. 11: Procedural model for applying integrated EPC-UML-modelling

tive business classes rather than technical objects or classes. Therefore the suffix 'requirement analysis' has been added.

- Owing to the integration of process orientation and object-orientation, the first steps in software development projects can be the same as in business process modelling projects, i.e. the step of requirements definition uses the same types of diagrams. The design phase can build on this information. More detailed class diagrams with a technical view can be derived from the requirement analysis class diagrams. The classes of the technical design class diagrams may be more or less the same as in the requirement analysis class diagrams with just more detailed operations and attributes, but due to the different objectives of the requirement analysis step and the design step, the class definitions can also differ extensively. As proposed by the UML recommendations, statechart diagrams, collaboration diagrams, and sequence diagrams can be used to emphasise essential subjects which have to be detailed. The information of EPCs can be used as input for that. Activity diagrams are not mentioned in the proposed procedural model since the information content of activity diagrams is already covered by the EPC method. But information can be converted between the two kinds of diagrams as shown before if desired. For implementation purposes, like defining dependencies between software components and the configuration of run-time processing elements, component diagrams and deployment diagrams are used.

The purpose of these procedural models is to explain the relationships between the diagrams and their roles in a project. They do not propose a straightforward sequence of requirements analysis step, design step, and implementation step as in a waterfall approach. Approaches with feedback mechanisms like prototyping and iterative loops do also fit to the procedual models.

8 References

Ambler, S. W.: What's Missing from the UML? Techniques that can help model effective business applications. Object Magazine 7(1997)8, (http://www.sigs.com/publications/objm/9710/ambler.html from 05-Mar-98).

Bungert, W.; Heß, H.: Objektorientierte Geschäftsprozeßmodellierung, in: IM Information Management, 10(1995)1, S. 52-63.

ESPRIT Consortium AMICE (ed.) Open System Architecture for CIM, Vol. 1, Berlin et al. 1989.

Ferstl, O. K.; Sinz, E. F.: Multi-layered Development of Business Process Models and Distributed Business Application Systems - An Object-Oriented Approach, in: König W., et al. (eds.): Distributed Information Systems in Business, New York et al. 1996, S. 159-179.

Fowler, M.; Scott, K.: UML Distilled - Applying the Standard Object Modeling Language, Reading et al. 1997.

Jacobson, I.; Christerson, M.; Jonsson, P.; Övergaard, G.: Object-Oriented Software Engineering - A Use Case Driven Approach, Wokingham et al. 1993.

Jacobson, I.; Ericsson, M.; Jacobson, A.: The Object Advantage, Business Process Reengineering with Object Technology, Wokingham et al. 1994.

Martin, J.: Information Engineering, Vol. I, II & III, London et al. 1989 & 1990.

Keller, G.; Nüttgens, M.; Scheer, A.-W.: Semantische Prozeßmodellierung auf der Grundlage "Ereignisgesteuerter Prozeßketten (EPK)", in: Publication of the Institut für Wirtschaftsinformatik, Paper 89, Saarbrücken 1992 (http://www.iwi.uni-sb.de/public/iwi-hefte/heft089.zip).

Nüttgens, M.; Feld, T.; Zimmermann, V.: Business Process Modeling with EPC and UML: Transformation or Integration?, in: Schader, M.; Korthaus, A. (Hrsg.): The Unified Modeling Language - Technical Aspects and Applications, Heidelberg 1998, pp. 250-261.

Olle, T. W.; Hagelstein, J.; MacDonald, I. G.; et al.: Information Systems Methodologies: a framework for understanding, Wokingham et al. 1988.

Österle, H..: Business in the Information Age: heading for new processes, Berlin et al. 1995.

Ovum (ed.): Analysts Give UML Guarded Welcome but Warn Users of Limitation, Jan 1998 (http://www.ovum.com/news/uml/umlpr.html from 26-Feb-98).

Ovum (ed.): UML White paper, 1997 (http://www.ovum.com/news/uml/umlwp.html from 26-Feb-98).

Rational Software Corporation et al. (1997a): UML Extension for Business Modeling, Version 1.1, Santa Clara 1997.

Rational Software Corporation et al. (1997b): UML Notation Guide, Version 1.1, Santa Clara 1997.

Rational Software Corporation et al. (1997c): UML Semantics, Version 1.1, Santa Clara 1997.

Scheer, A.-W.: ARIS - Business Process Frameworks, New York et al. 1998a (in preparation).

Scheer, A.-W.: ARIS - Business Process Modeling, New York et al. 1998b (in preparation).

Scheer, A.-W.; Nüttgens, M.; Zimmermann, V.: Objektorientierte Ereignisgesteuerte Prozeßkette (oEPK) - Methode und Anwendung, Publication of the Institut für Wirtschaftsinformatik, Paper 141, Saarbrücken 1997 (http://www.iwi.uni-sb.de/public/iwi-hefte/heft141.html).

Spurr, K.; Layzell, P.; Jennison, L.; Richards, N.: Business Objects: Software Solutions, Chichester et al. 1994.

Taylor, D. A.: Business Engineering with Object Technology, New York et al. 1995.

Die Veröffentlichungen des Instituts für Wirtschaftsinformatik (IWi) im Institut für empirische Wirtschaftsforschung an der Universität des Saarlandes erscheinen in unregelmäßiger Folge.

Heft 144: P. Loos, Th. Allweyer: Process Orientation and Object-Orientation - An Approach for Integrating UML and Event-Driven Process Chains (EPC), März 1998

Heft 142: Th. Allweyer, S. Leinenbach, A.-W. Scheer: Business Process Re-engineering in the Construction Industry, Oktober 1997

Heft 141: M. Nüttgens, V. Zimmermann, A.-W. Scheer: Objektorientierte Ereignisgesteuerte Prozeßkette (oEPK) - Methode und Anwendung -, Mai 1997

Heft 140: J. Sander, A.-W. Scheer: Offene Lernumgebungen in der Aus- und Weiterbildung am Beispiel des PPS-Trainers, März 1997

Heft 139: M. Bold, M. Hoffmann, A.-W. Scheer: Datenmodellierung für das Data Warehouse, März 1997

Heft 138: S. Stehle, A.-W. Scheer: Gestaltungsoptionen multimedialer Off- und Online- Lernsysteme aus pädagogischer Sicht, März 1997

Heft 137: M. Remme: Organisationsplanung durch konstruktivistische Modellierung, Februar 1997

Heft 136: M. Daneva, R. Heib, A.-W. Scheer: Benchmarking Business Process Models, Oktober 1996

Heft 135: M. Remme, J. Galler, M. Göbl, F. Habermann, A.-W. Scheer: IuK-Systeme für Planungsinseln, Oktober 1996

Heft 134: R. Heib, M. Daneva, A,-W.Scheer: Benchmarking as a Controlling Tool in Information Management, Oktober 1996

Heft 133: A.-W. Scheer: ARIS-House of Business Engineering, September 1996

Heft 132: J. Sander, A.-W. Scheer: Multimedia Engineering: Rahmenkonzept zum interdisziplinären Management von Multimedia-Projekten, Juli 1996

Heft 131: R. Heib, M. Daneva, A.-W. Scheer: ARIS-based Reference Model for Benchmarking, April 1996

Heft 130: R. Chen, V. Zimmermann, A.-W. Scheer: Geschäftsprozesse und integrierte Informationssysteme im Krankenhaus, April 1996

Heft 129: M. Nüttgens, V. Zimmermann, A.-W. Scheer: Business Process Reengineering in der Verwaltung, April 1996

Heft 128: P. Hirschmann, P. Lubiewski, A.-W. Scheer: Management von Konzernprozessen - Eine Fallstudie -, März 1996

Heft 127: J. Galler, M. Remme, A.-W. Scheer: Der Inseltrainer - Ein multimediales Lernsystem zur Qualifizierung in Planungsinseln, Januar 1996

Heft 126: P. Loos, O. Krier, P. Schimmel, A.-W. Scheer: WWW-gestützte überbetriebliche Logistik - Konzeption des Prototyps WODAN zur unternehmensübergreifenden Kopplung von Beschaffungs- und Vertriebssystemen, Februar 1996

Heft 125: M. Remme, A.-W. Scheer: Konstruktion von Prozeßmodellen, Februar 1996

Heft 124: M. Bold, E. Landwehr, A.-W. Scheer: Die Informations- und Kommunikationstechnologie als Enabler einer effizienten Verwaltungsorganisation, Februar 1996

Heft 123: P. Loos: Workflow und industrielle Produktionsprozesse - Ansätze zur Integration, Januar 1996

Heft 122: A.-W. Scheer: Industrialisierung der Dienstleistungen, Januar 1996

Heft 121: J. Galler: Metamodelle des Workflow-Managements, Dezember 1995

Heft 120: C. Kocian, F. Milius, M. Nüttgens, J. Sander, A.-W. Scheer: Kooperationsmodelle für vernetzte KMU-Strukturen, November 1995

Heft 119: W. Hoffmann, A.-W. Scheer, C. Hanebeck: Geschäftsprozeßmanagement in virtuellen Unternehmen, Oktober 1995

Heft 118: M. Remme, J. Galler, O. Gierhake, A.-W. Scheer: Die Erfassung der aktuellen Unternehmensprozesse als erste operative Phase für deren Re-engineering -Erfahrungsbericht-, September 1995

Heft 117: J. Galler, A.-W. Scheer, S. Peter: Workflow-Projekte: Erfahrungen aus Fallstudien und Vorgehensmodell, August 1995

Heft 116: A. Gücker, W. Hoffmann, M. Möbus, J. Moro, C. Troll: Objektorientierte Modellierung eines Qualitäts-informationssystems, Juni 1995

Heft 115: Th. Allweyer: Modellierung und Gestaltung adaptiver Geschäftsprozesse, Mai 1995

Heft 114: W. Hoffmann, A.-W. Scheer, M. Hoffmann: Überführung strukturierter Modellierungsmethoden in die Object Modeling Technique (OMT), März 1995

Heft 113: P. Hirschmann, A.-W. Scheer: Konzeption einer DV-Unterstützung für das überbetriebliche Prozeßmanagement, November 1994

Heft 112: A.-W. Scheer, M. Nüttgens, A. Graf v. d. Schulenburg: Informationsmanagement in deutschen Großunternehmen - Eine empirische Erhebung zu Entwicklungsstand und -tendenzen, November 1994

Heft 111: A.-W. Scheer: ARIS-Toolset: Die Geburt eines Softwareproduktes, Oktober 1994

Heft 110: M. Remme, A.-W. Scheer: Konzeption eines leistungskettenindizierten Informationssystemmanagements, September 1994

Heft 109: Th. Allweyer, P. Loos, A.-W. Scheer: An Empirical Study on Scheduling in the Process Industries, July 1994

Heft 108: J. Galler, A.-W. Scheer: Workflow-Management: Die ARIS-Architektur als Basis eines multimedialen Workflow-Systems, Mai 1994

Heft 107: R. Chen, A.-W. Scheer: Modellierung von Prozeßketten mittels Petri-Netz-Theorie, Februar 1994

Heft 106: W. Hoffmann; R. Wein; A.-W. Scheer: Konzeption eines Steuerungsmodells für Informationssysteme - Basis für die Real-Time-Erweiterung der EPK (rEPK), Dezember 1993

Heft 105: A. Hars; V. Zimmermann; A.-W. Scheer: Entwicklungslinien für die computergestützte Modellierung von Aufbau- und Ablauforganisation, Dezember 1993

Heft 104: A. Traut; T. Geib; A.-W. Scheer: Sichtgeführter Montagevorgang - Planung, Realisierung, Prozeßmodell, Juni 1993

Heft 103: wird noch nicht verlegt

Heft 102: P. Loos: Konzeption einer graphischen Rezeptverwaltung und deren Integration in eine CIP-Umgebung - Teil 1, Juni 1993

Heft 101: W. Hoffmann, J. Kirsch, A.-W. Scheer: Modellierung mit Ereignisgesteuerten Prozeßketten (Methodenbuch, Stand: Dezember 1992), Januar 1993

Heft 100: P. Loos: Representation of Data Structures Using the Entity Relationship Model and the Transformation in Relational Databases, January 1993

Heft 99: H. Heß: Gestaltungsrichtlinien zur objektorientierten Modellierung, Dezember 1992

Heft 98: R. Heib: Konzeption für ein computergestütztes IS-Controlling, Dezember 1992

Heft 97: Chr. Kruse, M. Gregor: Integrierte Simulationsmodellierung in der Fertigungssteuerung am Beispiel des CIM-TTZ Saarbrücken, Dezember 1992

Heft 96: P. Loos: Die Semantik eines erweiterten Entity-Relationship-Modells und die Überführung in SQL-Datenbanken, November 1992

Heft 95: R. Backes, W. Hoffmann, A.-W. Scheer: Konzeption eines Ereignisklassifikationssystems in Prozeßketten, November 1992

Heft 94: Chr. Kruse, A.-W. Scheer: Modellierung und Analyse dynamischen Systemverhaltens, Oktober 1992

Heft 93: M. Nüttgens, A.-W. Scheer, M. Schwab: Integrierte Entsorgungssicherung als Bestandteil des betrieblichen Informations-managements, August 1992

Heft 92: A. Hars, R. Heib, Chr. Kruse, J. Michely, A.-W. Scheer: Approach to classification for information engineering - methodology and tool specification, August 1992

Heft 91: C. Berkau: Konzept eines controllingbasierten Prozeßmanagers als intelligentes Multi-Agent-System, Januar 1992

Heft 90: C. Berkau, A.-W. Scheer: VOKAL (System zur Vorgangskettendarstellung), Teil 2: VKD-Modellierung mit Vokal, Dezember 1991 (wird nicht verlegt)

Heft 89: G. Keller, M. Nüttgens, A.-W. Scheer: Semantische Prozeßmodellierung auf der Grundlage "Ereignisgesteuerter Prozeßketten (EPK)", Januar 1992

Heft 88: W. Hoffmann, B. Maldener, M. Nüttgens, A.-W. Scheer: Das Integrationskonzept am CIM-TTZ Saarbrücken (Teil 2: Produktionssteuerung), Januar 1992

Heft 87: M. Nüttgens, G. Keller, S. Stehle: Konzeption hyperbasierter Informationssysteme, Dezember 1991

Heft 86: A.-W. Scheer: Koordinierte Planungsinseln: Ein neuer Lösungsansatz für die Produktionsplanung, November 1991

Heft 85: W. Hoffmann, M. Nüttgens, A.-W. Scheer, St. Scholz: Das Integrationskonzept am CIM-TTZ Saarbrücken (Teil 1: Produktionsplanung), Oktober 1991

Heft 84: A. Hars, R. Heib, Ch. Kruse, J. Michely, A.-W. Scheer: Concepts of Current Data Modelling Methodologies - A Survey - 1991

Heft 83: A. Hars, R. Heib, Ch. Kruse, J. Michely, A.-W. Scheer: Concepts of Current Data Modelling Methodologies - Theoretical Foundations - 1991

Heft 82: C. Berkau: VOKAL (System zur Vorgangskettendarstellung und -analyse), Teil 1: Struktur der Modellierungsmethode - Dezember 1991 (wird nicht verlegt)

Heft 81: A.-W. Scheer: Papierlose Beratung - Werkzeugunterstützung bei der DV-Beratung, August 1991

Heft 80: G. Keller, J. Kirsch, M. Nüttgens, A.-W. Scheer: Informationsmodellierung in der Fertigungssteuerung, August 1991

Heft 79: A.-W. Scheer: Konsequenzen für die Betriebswirtschaftslehre aus der Entwicklung der Informations- und Kommunikationstechnologien, Mai 1991

Heft 78: H. Heß: Vergleich von Methoden zum objektorientierten Design von Softwaresystemen, August 1991

Heft 77: W. Kraemer: Ausgewählte Aspekte zum Stand der EDV-Unterstützung für das Kostenmanagement: Modellierung benutzerindividueller Auswertungssichten in einem wissensbasierten Controlling-Leitstand, Mai 1991

Heft 76: Ch. Houy, J. Klein: Die Vernetzungsstrategie des Instituts für Wirtschaftsinformatik - Migration vom PC-Netzwerk zum Wide Area Network (noch nicht veröffentlicht)

Heft 75: M. Nüttgens, St. Eichacker, A.-W. Scheer: CIM-Qualifizierungskonzept für Klein- und Mittelunternehmen (KMU), Januar 1991

Heft 74: R. Bartels, A.-W. Scheer: Ein Gruppenkonzept zur CIM-Einführung, Januar 1991

Heft 73: A.-W. Scheer, M. Bock, R. Bock: Expertensystem zur konstruktionsbegleitenden Kalkulation, November 1990

Heft 72: M. Zell: Datenmanagement simulationsgestützter Entscheidungsprozesse am Beispiel der Fertigungssteuerung, November 1990

Heft 71: D. Aue, M. Baresch, G. Keller: **URMEL**, Ein **U**nte**R**nehmens**M**od**EL**lierungsansatz, Oktober 1990

Heft 70: St. Spang, K. Ibach: Zum Entwicklungsstand von Marketing-Informationssystemen in der Bundesrepublik Deutschland, September 1990

Heft 69: A.-W. Scheer, R. Bartels, G. Keller: Konzeption zur personalorientierten CIM-Einführung, April 1990

Heft 68: W. Kraemer: Einsatzmöglichkeiten von Expertensystemen in betriebswirtschaftlichen Anwendungsgebieten, März 1990

Heft 67: A.-W. Scheer: Modellierung betriebswirtschaftlicher Informationssysteme (Teil 1: Logisches Informationsmodell), März 1990

Heft 66: W. Jost, G. Keller, A.-W. Scheer: CIMAN - Konzeption eines DV-Tools zur Gestaltung einer CIM-orientierten Unternehmensarchitektur, März 1990

Heft 65: A. Hars, A.-W. Scheer: Entwicklungsstand von Leitständen[1], Dezember 1989

Heft 64: C. Berkau, W. Kraemer, A.-W. Scheer: Strategische CIM-Konzeption durch Eigenentwicklung von CIM-Modulen und Einsatz von Standardsoftware, Dezember 1989

Heft 63: A.-W. Scheer: Unternehmens-Datenbanken - Der Weg zu bereichsübergreifenden Datenstrukturen, September 1989

Heft 62: M. Zell, A.-W. Scheer: Simulation als Entscheidungsunterstützungsinstrument in CIM, September 1989

Heft 61: A.-W. Scheer, G. Keller, R. Bartels: Organisatorische Konsequenzen des Einsatzes von Computer Aided Design (CAD) im Rahmen von CIM, Januar 1989

Heft 60: A.-W. Scheer, W. Kraemer: Konzeption und Realisierung eines Expertenunterstützungssystems im Controlling, Januar 1989

Heft 59: R. Herterich, M. Zell: Interaktive Fertigungssteuerung teilautonomer Bereiche, November 1988

Heft 58: A.-W. Scheer: CIM in den USA - Stand der Forschung, Entwicklung und Anwendung, November 1988

Heft 57: A.-W. Scheer: Present Trends of the CIM Implementation (A qualitative Survey) Juli 1988

Heft 56: A.-W. Scheer: Enterprise wide Data Model (EDM) as a Basis for Integrated Information Systems, Juli 1988

Heft 55: D. Steinmann: Expertensysteme (ES) in der Produktionsplanung und -steuerung (PPS) unter CIM-Aspekten, November 1987, Vortrag anläßlich der Fachtagung "Expertensysteme in der Produktion" am 16. und 17.11.1987 in München

Heft 54: U. Leismann, E. Sick: Konzeption eines Bildschirmtext-gestützten Warenwirtschaftssystems zur Kommunikation in verzweigten Handelsunternehmungen, August 1986

Heft 53: A.-W. Scheer: Neue Architektur für EDV-Systeme zur Produktionsplanung und -steuerung, Juli 1986

Heft 52: P. Loos, T. Ruffing: Verteilte Produktionsplanung und -steuerung unter Einsatz von Mikrocomputern, Juni 1986

Heft 51: A.-W. Scheer: Strategie zur Entwicklung eines CIM-Konzeptes - Organisatorische Entscheidungen bei der CIM-Implementierung, Mai 1986

Heft 50: A.-W. Scheer: Konstruktionsbegleitende Kalkulation in CIM-Systemen, August 1985

Heft 49: A.-W. Scheer: Wirtschaftlichkeitsfaktoren EDV-orientierter betriebswirtschaftlicher Problemlösungen, Juni 1985

Heft 48: A.-W. Scheer: Kriterien für die Aufgabenverteilung in Mikro-Mainframe Anwendungssystemen, April 1985

Heft 47: A.-W. Scheer: Integration des Personal Computers in EDV-Systeme zur Kostenrechnung, August 1984

Heft 46: H. Krcmar: Die Gestaltung von Computer am-Arbeitsplatz-Systemen - ablauforientierte Planung durch Simulation, August 1984

Heft 45: J. Ahlers, W. Emmerich, H. Krcmar, A. Pocsay, A.-W. Scheer, D. Siebert: EPSOS-D, Ein Werkzeug zur Messung der Qualität von Software-Systemen, August 1984

Heft 44: A.-W. Scheer: Schnittstellen zwischen betriebswirtschaftlicher und technische Datenverarbeitung in der Fabrik der Zukunft, Juli 1984

Heft 43: A.-W. Scheer: Einführungsstrategie für ein betriebliches Personal-Computer-Konzept, März 1984

Heft 42: A.-W. Scheer: Factory of the Future, Vorträge im Fachausschuß "Informatik in Produktion und Materialwirtschaft" der Gesellschaft für Informatik e. V., Dezember 1983

Heft 41: H. Krcmar: Schnittstellenprobleme EDV-gestützter Systeme des Rechnungswesens, August 1983, Vortrag anläßlich der 4. Saarbrücker Arbeitstagung "Rechnungswesen und EDV" in Saarbrücken vom 26. - 28.09.1983

Heft 40: A.-W. Scheer: Strategische Entscheidungen bei der Gestaltung EDV-gestützter Systeme des Rechnungswesens, August 1983, Vortrag anläßlich der 4. Saarbrücker Arbeitstagung "Rechnungswesen und EDV" in Saarbrücken vom
26. - 28.09.1983

Heft 39: A.-W. Scheer: Personal Computing - EDV-Einsatz in Fachabteilungen, Juni 1983

Heft 38: A.-W. Scheer: Interaktive Methodenbanken: Benutzerfreundliche Datenanalyse in der Marktforschung, Mai 1983

Heft 37: A.-W. Scheer: DV-gestützte Planungs- und Informationssysteme im Produktionsbereich, September 1982

Heft 36: A.-W. Scheer: Rationalisierungserfolge durch Einsatz der EDV - Ziel und Wirklichkeit, August 1982, Vortrag anläßlich der 3. Saarbrücker Arbeitstagung "Rationalisierung" in Saarbrücken vom 04. - 06. 10.1982

Heft 35: J. Ahlers, W. Emmerich, H. Krcmar, A. Pocsay, A.-W. Scheer, D. Siebert: EPSOS-D, Konzept einer computergestützten Prüfungsumgebung, Juli 1982

Heft 34: J. Ahlers, W. Emmerich, H. Krcmar, A. Pocsay, A.-W. Scheer, D. Siebert: EPSOS - Ein Ansatz zur Entwicklung prüfungsgerechter Software-Systeme, Mai 1982

Heft 33: A.-W. Scheer: Disposition- und Bestellwesen als Baustein zu integrierten Warenwirtschaftssystemen, März 1982, Vortrag anläßlich des gdi-Seminars "Integrierte Warenwirtschafts-Systeme" in Zürich vom 10. - 12. Dezember 1981

Heft 32: A.-W. Scheer: Einfluß neuer Informationstechnologien auf Methoden und Konzepte der Unternehmensplanung, März 1982, Vortrag anläßlich des Anwendergespräches "Unternehmensplanung und Steuerung in den 80er Jahren in Hamburg vom 24. - 25.11.1981

Die Hefte 1 - 31 werden nicht mehr verlegt.