CHAPTER 9
Enterprise Modeling

August-Wilhelm Scheer
Frank Habermann
Oliver Thomas
Saarland University

1. MODELING BASICS 280
1.1. Creating a Model World 280
1.2. Levels of Abstraction 281
1.3. Principles of Modeling 283
1.3.1. Principle of Correctness 284
1.3.2. Principle of Relevance 284
1.3.3. Principle of Cost vs. Benefit 284
1.3.4. Principle of Clarity 284
1.3.5. Principle of Comparability 284
1.3.6. Principle of Systematic Structure 284
2. MODELING BENEFITS 284
2.1. Benefits for Business Administration and Organizational Processes 284
2.2. Benefits for Developing Information Systems 285
3. MODELING VIEWS AND METHODS 286
3.1. Process-Oriented Enterprise Modeling 286
3.1.1. Organization Views 286
3.1.2. Function Views 287
3.1.3. Output Views 287
3.1.4. Data Views 288
3.1.5. Process View 290
3.2. Object-Oriented Enterprise Modeling 291
4. MODELING ARCHITECTURES 293
4.1. Architecture of Integrated Information Systems (ARIS) 293
4.2. Information Systems Methodology (IFIP-ISM) 300
4.3. CIM Open System Architecture (CIMOSA) 301
4.4. Zachman Framework 302
5. MODELING TOOLS 303
5.1. Benefits of Computerized Enterprise Modeling 303
5.2. Characterization of Modeling Tools 303
6. MODELING OUTLOOK 306
REFERENCES 307
ADDITIONAL READING 307

1. MODELING BASICS

1.1. Creating a Model World

Enterprises are sociotechnological real-world systems. A system is a composition of elements and their relationships. Examples of elements of the system enterprise are employees, products, activities, machines, computers, and software, which interact in manifold ways. They can be described by their structures (static view) as well as their behavior (dynamic view). Because enterprises are very complex, in many cases analysis cannot be done directly on the real-world-system.

Enterprise modeling aims at reducing the complexity of the system to be studied. Only specific aspects of an enterprise are examined, such as data structures, input–output relationships, and logistic processes. Such subsystems of the real world are called "mini world" or "model domain." Thus, models are descriptions of the most important characteristics of a focused domain. The creative process of building such an abstracted picture of a real-world system is called modeling (see Figure 1).

In addition to the actual purpose of modeling, the modeling methods applied also determine the focus of the model. This is particularly true when a certain type of method has already been defined. Models reproduce excerpts of reality. They are created by abstracting the properties of real objects, whereas their essential structures and behavior remain intact (homomorphy). Not only the content-related purpose of the model but also the permissible illustration methods determine to what extent nonessential characteristics may be abstracted. For example, if an object-oriented modeling approach or a system-theoretic approach is selected, modeling only leads to objects applicable to the syntax or the semantics of these particular methods. In order to avoid lock-in by certain methods, architectures and methodologies are developed independently of any particular method, while supporting a generic business process definition.

The following sections we will discuss modeling methods and architectures in greater detail.

1.2. Levels of Abstraction

Abstraction is the basic concept of modeling. In system theory, we know many perspectives of abstraction. Thus, we can concentrate either on the structure or behavior of enterprises; on certain elements of the enterprise, such as data, employees, software, products; or on a bounded domain, such as sales, accounting, manufacturing. Typically, in enterprise modeling projects, several of these perspectives are combined. For instance, a typical modeling project might be "Describe the sales data structures."

However, the term *level of abstraction* describes the relationships between a model system and the respective real-world system. That is, it describes the steps of abstraction. From low to high abstraction we can distinguish at least three levels in modeling: instances, application classes, and meta classes.

At the instance level, each single element of the real world is represented through one single element in our model. For example, in the case of modeling an individual business process, every element involved in the business process is instantiated by the affixed name, such as customer (no.) 3842, material M 32, or completed order O 4711, etc. (see Figure 2).

Enterprise modeling at the instance level is used for controlling individual business processes. In manufacturing, this is for creating work schedules as the manufacturing process descriptions for individual parts or manufacturing orders. In office management, individual business processes are executed through workflow systems. Therefore, they must have access to information regarding the respective control structure and responsible entities or devices for every single business case.

At the application class level, we abstract from instance properties and define classes of similar entities. For example, all individual customers make up the class "customer," all instances of orders constitute the class "order," and so on (see Figure 2). Every class is characterized by its name and the enumeration of its attributes, by which the instance is described. For example, the class "cus-

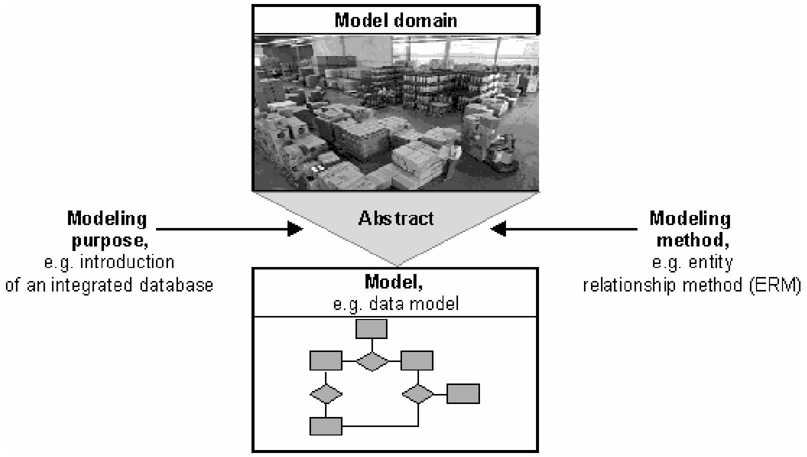

Model domain
Abstract
Modeling purpose, e.g. introduction of an integrated database
Modeling method, e.g. entity relationship method (ERM)
Model, e.g. data model

Figure 1 Modeling.

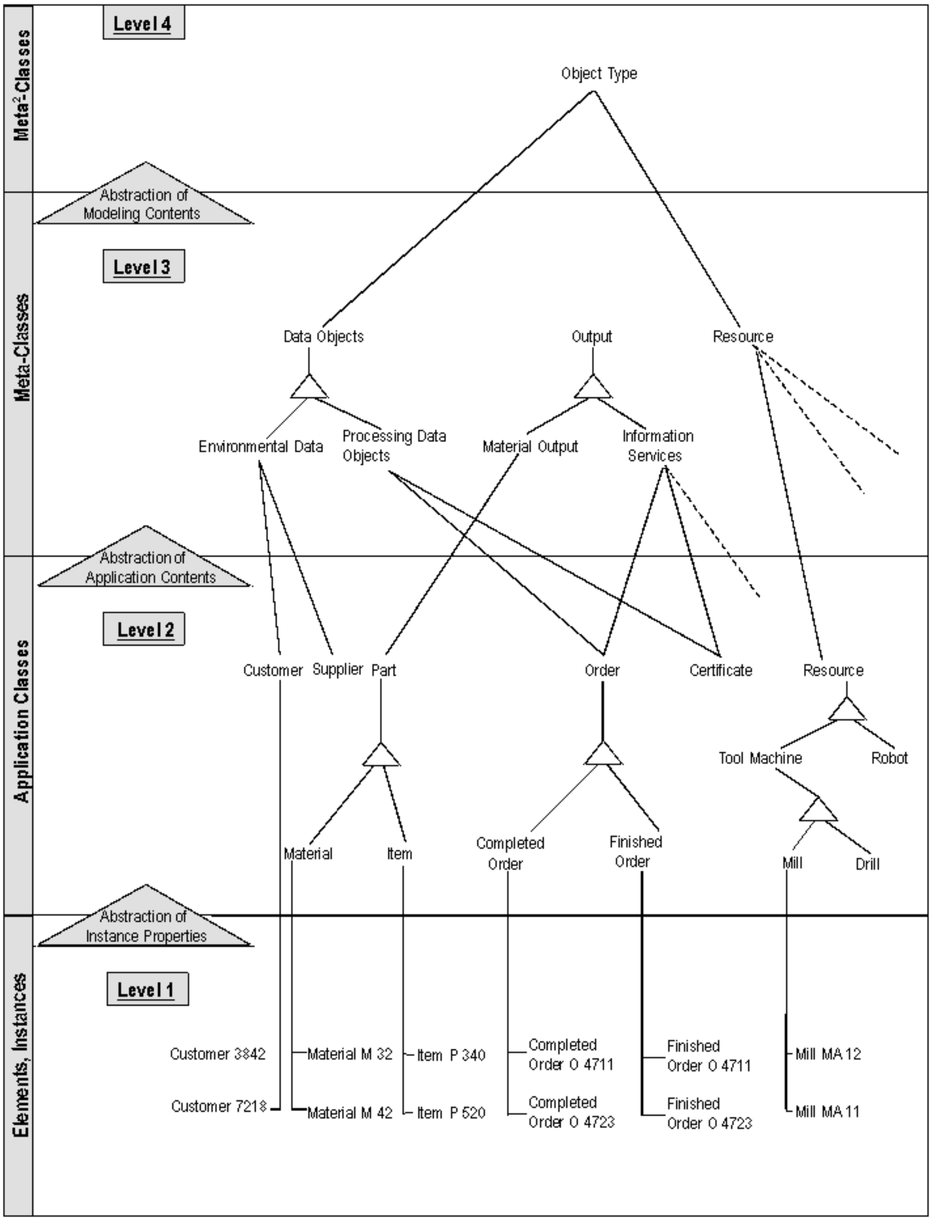


Figure 2 Levels of Abstraction.

tomer" is characterized by the attributes customer number, customer name, and payment period. The instances of these characteristics are the focus of the description at Level 1.

Finding classes is always a creative task. It thus depends on the subjective perspective of the modeler. Therefore, when defining, for example, order designations, we will only abstract specific properties of cases 4711 or 4723, respectively, leading to the classes "completed order" or "finished order." At Level 2, we will abstract the "completed" and "finished" properties and create the parent class "order" from the subset. This operation is known as generalization and is illustrated by a triangular symbol.

When quantities are generalized, they are grouped to parent quantities. This makes order instances of Level 1 instances of the class "order" as well. The class "order" is designated as the property "order status," making it possible to allocate the process state "completed" or "finished" to every instance. Materials and items are also generalized, making them "parts" and "resources."

Thus, Level 2 contains application-related classes of enterprise descriptions. On the other hand, with new classes created from similar classes of Level 2 by abstracting their application relationships, these are allocated to Level 3, the meta level, as illustrated in Figure 2. Level 2 classes then become instances of these meta classes. For example, the class "material output" contains the instances "material" and "item" as well as the generalized designation "part." The class "information services" contains the class designation "order," along with its two child designations, and the class designation "certificate." The creation of this class is also a function of its purpose. Thus, either the generalized classes of Level 2 or their subclasses can be included as elements of the meta classes.

When classes are created, overlapping does not have to be avoided at all costs. For example, from an output flow point of view, it is possible to create the class "information services" from the classes "order" and "certificate." Conversely, from the data point of view, these are also data objects, making them instances of the class "data objects" as well.

The classes at modeling Level 3 define every object necessary for describing the facts at Level 2. These objects make up the building blocks for describing the applications at Level 2. On the other hand, because the classes at Level 2 comprise the terminology at Level 1, objects at Level 3 are also the framework for describing instances.

This abstraction process can be continued by once again grouping the classes at Level 3 into classes that are then allocated to the meta^2 level. Next, the content-related modeling views are abstracted. In Figure 2, the general class "object type" is created, containing all the meta classes as instances.

1.3. Principles of Modeling

Enterprise modeling is a creative process and can therefore not be completely directed by rules. However, if certain standards are observed, it is indeed possible to classify and understand third-party models. Furthermore, it is also a good idea to establish certain quality standards for enterprise models.

Figure 3 illustrates the relationship between principles and enterprise modeling. A principle is a fundamental rule or strategy that guides the entire process of enterprise modeling. Thus, modeling

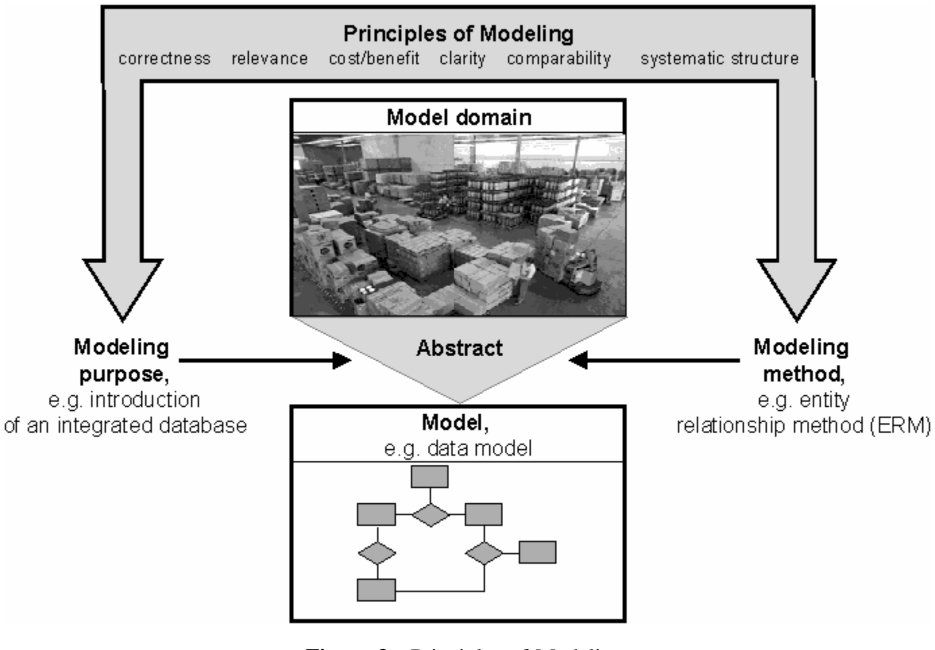


Figure 3 Principles of Modeling.

principles concern the association between the domain and the respective model as well as the selection of appropriate modeling methods and tool support. A modeling method is a set of components and a description of how to use these components in order to build a model. A modeling tool is a piece of software that can be used to support the application of a method. We will discuss modeling methods and tools in more detail in the following chapters.

1.3.1. Principle of Correctness

The correctness of models depends on correct semantics and syntax, that is, whether syntax of the respective metamodel is complete and consistent. Semantic correctness of a model is measured by how closely it complies with the structure and behavior of the respective object system. In real-world applications, compliance with these requirements can be proven only after simulation studies have been carried out or other similar efforts have been made. Some modeling tools provide a simulation function that can be used for this purpose.

1.3.2. Principle of Relevance

Excerpts of the real-world object system should only be modeled provided they correspond with the purpose of the model. Models should not contain more information than necessary, thus keeping the cost vs. benefit ratio down to an acceptable level.

1.3.3. Principle of Cost vs. Benefit

One of the key factors ensuring a good cost vs. benefit ratio is the amount of effort necessary to create the model, the usefulness of modeling the scenario, and how long the model will be used.

1.3.4. Principle of Clarity

"Clarity" ensures that a model is understandable and usable. It also determines how pragmatic the relationship between the model and the user is. Because models contain a large amount of information regarding technical and organizational issues, only specialists are usually able to understand them quickly. Once models are broken down into subviews, individual views are easier to comprehend.

1.3.5. Principle of Comparability

Models created in accordance with a consistent conceptual framework and modeling methods are comparable if the objects have been named in conformity with established conventions and if identical modeling objects as well as equivalent degrees of detailing have been used. In models created with different modeling methods, it is important to make sure that their metamodels can be compared.

1.3.6. Principle of Systematic Structure

This principle stipulates that it should be possible to integrate models developed in various views, such as data models, organizational models, and business process models. This requires an integrated methodology, that is, an architecture providing a holistic metamodel (see Section 4).

2. MODELING BENEFITS

Enterprise models are used as instruments for better understanding business structures and processes. Thus, the general benefit of enterprise modeling is in business administration and organizational processes. Focusing computer support in business, we can use enterprise models additionally to describe the organizational impacts of information technology. Consequently, the second major benefit of enterprise modeling is for developing information systems.

2.1. Benefits for Business Administration and Organizational Processes

Corporate mission statements entail the production and utilization of material output and services by combining production factors. Multiple entities and devices are responsible for fulfilling these tasks. In accordance with corporate objectives, their close collaboration must be ensured. In order for human beings to be able to handle complex social structures such as enterprises, these structures must be broken down into manageable units. The rules required for this process are referred to as "organization."

Structural or hierarchical organizations are characterized by time-independent (static) rules, such as by hierarchies or enterprise topologies. Here, relationships involving management, output, information, or communication technology between departments, just to name a few, are entered. Org charts are some of the simple models used to depict these relationships.

Process organizations, on the other hand, deal with time-dependent and logical (dynamic) behavior of the processes necessary to complete the corporate mission. Hierarchical and process organizations are closely correlated. Hierarchical organizations have been a key topic in business theory for years.

However, due to buzzwords such as business process reengineering (BPR), process organizations have moved into the spotlight in recent years.

Reasons for creating business process models include:

- Optimizing organizational changes, a byproduct of BPR
- Storing corporate knowledge, such as in reference models
- Utilizing process documentation for ISO-9000 and other certifications
- Calculating the cost of business processes
- Leveraging process information to implement and customize standard software solutions or workflow systems (see Section 2.2).

Within these categories, other goals can be established for the modeling methods. In business process improvement, we must therefore identify the components that need to be addressed. Some of the many issues that can be addressed by business process optimization are:

- Changing the process structure by introducing simultaneous tasks, avoiding cycles, and streamlining the structure
- Changing organizational reporting structures and developing employee qualification by improving processing in its entirety
- Reducing the amount of documentation, streamlining and accelerating document and data flow
- Discussing possible outsourcing measures (shifting from internal to external output creation)
- Implementing new production and IT resources to improve processing functions

In these examples, we are referring to numerous modeling aspects, such as process structures, hierarchical organizations, employee qualification, documents (data), and external or internal output as well as production and IT resources. Obviously, an enterprise model, particularly a business process model for the purpose of optimization, must be fairly complex. Moreover, it should address multiple aspects, for which numerous description methods are necessary. These various purposes determine the kind of modeling objects as well as the required granularity.

2.2. Benefits for Developing Information Systems

Information systems can be designed as custom applications or purchased as off-the-shelf standard solutions. After the initial popularity of custom applications, integrated standard solutions are now the norm. With the advent of new types of software, such as componentware (where software components for certain application cases are assembled to form entire applications), a blend between the two approaches has recently been making inroads.

The development of custom applications is generally expensive and is often plagued by uncertainties, such as the duration of the development cycle or the difficulty of assessing costs. Thus, the tendency to shift software development from individual development to an organizational form of industrial manufacturing—in "software factories"—is not surprising.

In this context, multiple methods for supporting the software development process have been developed. They differ according to their focus on the various software development processes and their preferred approach regarding the issue at hand, such as data, event, or function orientation, respectively.

Due to the wide range of methods that differ only slightly from one another, this market is cluttered. In fact, the multitude of products and approaches has actually impeded the development of computer-aided tools based on these methods. We therefore recommend a methodology (study of methods) covering various development methods. The following are typical questions that leverage the framework capabilities of a methodology:

1. Are there really so many totally different ways of designing a computer-aided information system?
2. If not, how similar are these methods? If so, why are there so many different ways?
3. Is there an optimal way of developing an information system?
4. Where does the development process start and where does it end?
5. What does the finished product of the design process look like?
6. How many steps are necessary to obtain a development result?
7. Should only one particular kind of information system be used or are several methods required, each for a different system? According to which criteria should the methods be selected?

The purpose of these questions is to classify and evaluate the various modeling methods. After these issues are addressed, there is, however, a second group of reasons for dealing with information system design methodologies (ISDMs), resulting from the fact that usually several business partners are involved in complex development projects. Sometimes they use different development methods, or the results of their work might overlap. Only an architecture integrating the individual methods, confirming agreement or pointing out any overlap, can lead to mutual understanding.

The alternative to individual software development is to buy a standardized business administration software solution. Such solutions include modules for accounting, purchasing, sales, production planning, and so on. Financial information systems are characterized by a markedly high degree of complexity. Many corporate and external business partners are involved in the implementation of information systems. This becomes apparent in light of seamlessly integrated data processing, where data is shared by multiple applications. Examples include comprehensive IS-oriented concepts implemented in enterprises, CIM in manufacturing companies, IS-supported merchandise management systems for retailers, and electronic banking in financial institutions.

Until the mid-1990s, the ratio between the effort of implementing financial packaged applications in organizations and their purchase price was frequently more than 5:1. This ratio is so high because off-the-shelf systems are more or less easy to install, yet users must also determine which goals (strategies) they wish to reach with the system, how the functionality of the system can achieve this, and how to customize, configure, and technically implement the package.

With hardware and software costs rapidly decreasing, that ratio became even worse. Small and medium-sized enterprises (SMEs) are not able to pay consultants millions of dollars for implementation. Hence, architectures, methods, and tools have become increasingly popular because they can help reduce the cost of software implementation and at the same time increase user acceptance of standard software solutions.

Several modeling approaches are possible:

- Reduce the effort necessary for creating the target concept by leveraging "best-practice case" knowledge available in reference models.
- Create a requirements definition by leveraging modeling techniques to detail the description.
- Document the requirements definition of the standard software by means of semantic modeling methods, making the business logic more understandable.
- Use semantic models to automate reconciliation of the requirements definition of the target concept with the standard software as much as possible, cutting down on the need for specific IS skills.
- Leverage semantic models as a starting point for maximum automation of system and configuration customizing.

3. MODELING VIEWS AND METHODS

3.1. Process-Oriented Enterprise Modeling

Generally speaking, a business process is a continuous series of enterprise activities, undertaken for the purpose of creating output. The starting point and final product of the business process are the output requested and utilized by corporate or external customers. Business processes often enable the value chain of the enterprise as well as focusing on the customer when the output is created.

In the following, we explain the key issues in modeling business processes with a simple example from customer order processing. First, let us outline the scenario:

A customer wants to order several items that need to be manufactured. Based on customer and item information, the feasibility of manufacturing this item is studied. Once the order has arrived, the necessary materials are obtained from a supplier. After arrival of the material and subsequent order planning, the items are manufactured according to a work schedule and shipped to the customer along with the appropriate documentation.

This scenario will be discussed from various points of view. As we have already seen, in system theory we can distinguish between system structures and system behavior. We will begin by describing the responsible entities and relationships involved in the business process. Then, by means of function flow, we will describe the dynamic behavior. Output flows describe the results of executing the process, and information flows illustrate the interchange of documents involved in the process.

Functions, output producers (organizational units), output, and information objects are illustrated by various symbols. Flows are depicted by arrows.

3.1.1. Organization Views

Figure 4 depicts the responsible entities (organizational units) involved in the business process, along with their output and communication relationships, illustrated as context or interaction diagrams. The

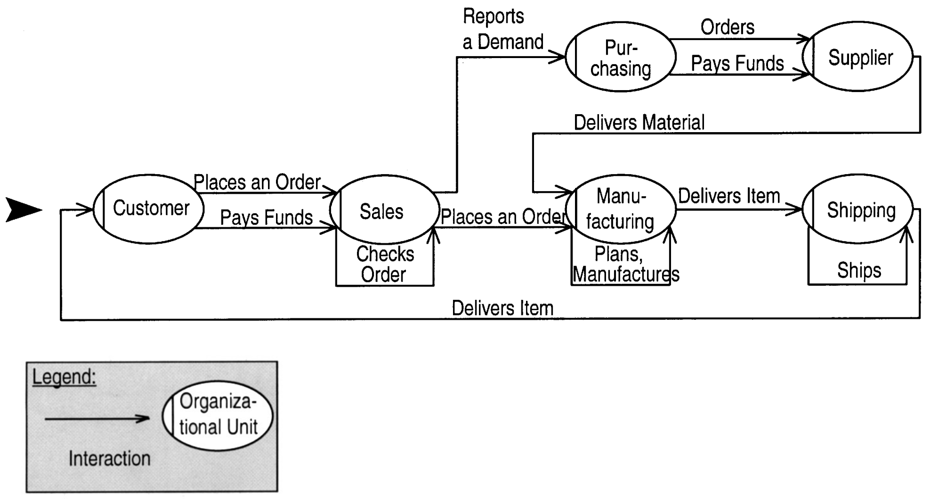


Figure 4 Interaction Diagram of the Business Process "Order Processing."

sequence in which processes are carried out is not apparent. Nevertheless, this provides an initial view of the business process structure. In complex processes, the myriad interchanges among the various business partners can become somewhat confusing. In addition to the various interactions, it is also possible to enter the activities of the responsible entities. This has been done in only a few places.

The class of organization views also includes the hierarchical organization structure (org charts). Org charts are created in order to group responsible entities or devices that are executing the same work object. This is why the responsible entities, responsible devices, financial resources, and computer hardware are all assigned together to the organization views.

3.1.2. Function Views

Figure 5 describes the same business process by depicting the activities (functions) to be executed, as well as their sequence. The main issue is not responsible entities, as with the interaction diagram, but rather the dynamic sequence of activities. For illustration purposes, the organizational units are also depicted in Figure 5. Due to redundancies, their interrelationship with the interaction diagram is not as obvious. As function sequences for creating output, function flows characterize the business process. The output flows themselves will be displayed individually.

The class of function views also includes the hierarchical structure of business activities transforming input into output. According to the level of detail, they are labeled "business processes," "processes," "functions," and "elementary functions."

Because functions support goals, yet are controlled by them as well, goals are also allocated to function views—because of the close linkage. In application software, computer-aided processing rules of a function are defined. Thus, application software is closely aligned with "functions" and is also allocated to function views.

3.1.3. Output Views

The designation "output" is very heterogeneous. Business output is the result of a production process, in the most general sense of the word. Output can be physical (material output) or nonphysical (services). Whereas material output is easily defined, such as by the delivery of material, manufactured parts, or even the finished product, the term *services* is more difficult to define because it comprises heterogeneous services, such as insurance services, financial services, and information brokering services. Figure 6 illustrates this simplified classification of "output"—that is, also "input"—as a hierarchical diagram.

Concerning our business process example, the result of the function "manufacture item" in Figure 7 is the material output, defined by the manufactured item. Likewise, quality checks are carried out and documented during the manufacturing process. All data pertinent to the customer are captured in "order documents," themselves a service by means of the information they provide. After every intercompany function, output describing the deliverable is defined, which in turn is entering the next

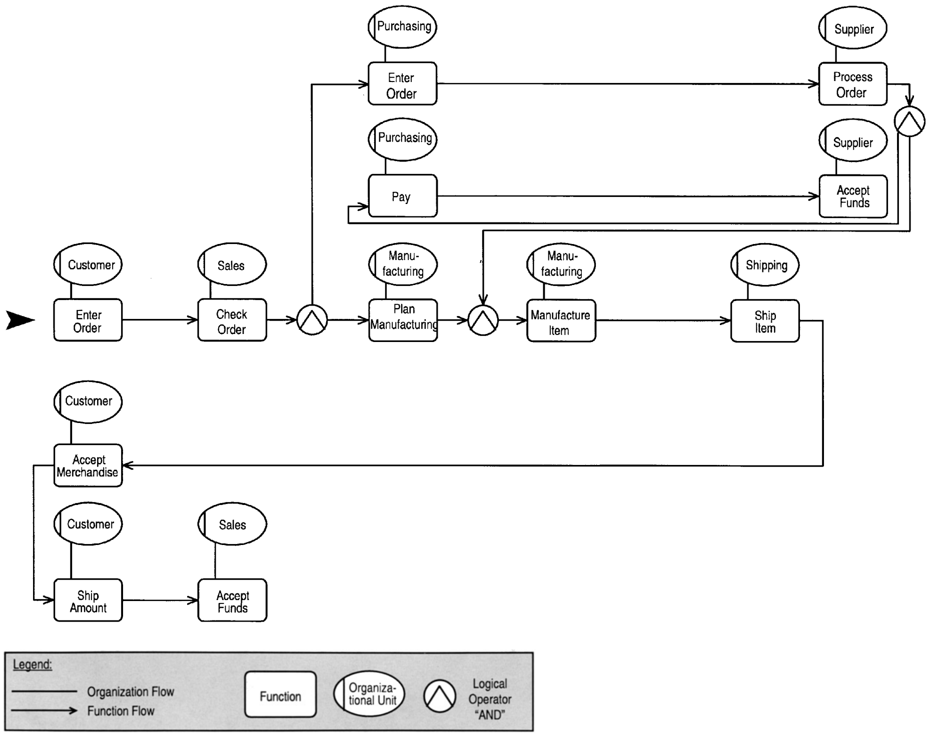


Figure 5 Function Flow of the Business Process "Order Processing."

process as input. To avoid cluttering the diagram, the organizational units involved are not depicted. It is not possible to uniquely derive the function sequence from the illustration of the output flow.

3.1.4. Data Views

The designations "data" and "information" are used synonymously. In addition to information services, other information, used as environment descriptions during the business processes, constitutes

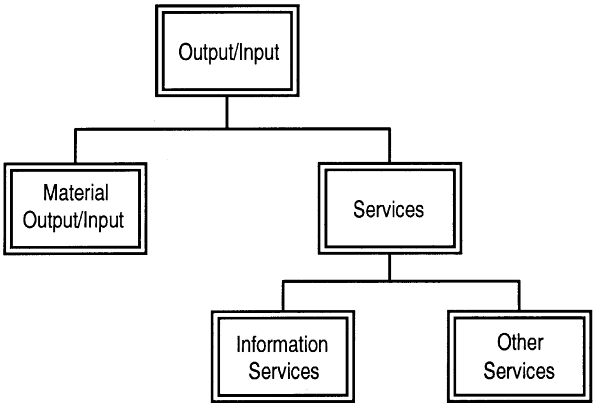


Figure 6 Types of Input–Output.

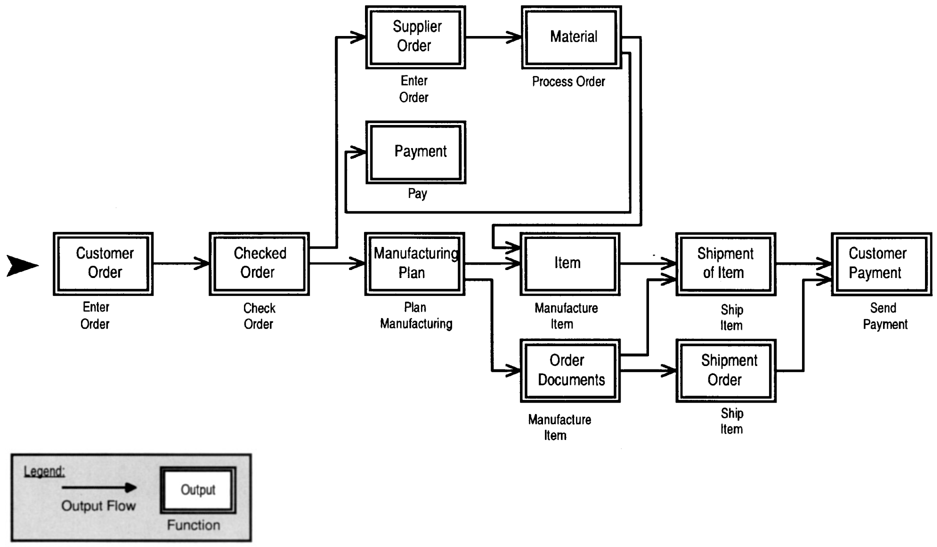


Figure 7 Output Flow of the Business Process "Order Processing."

process components. Figure 8 illustrates the information objects of our business process example, along with the data interchanged among them. Objects listed as information services have double borders. Information objects describing the environment of the business process are shown as well, such as data regarding suppliers, items, or work schedules. These data are necessary to create infor-

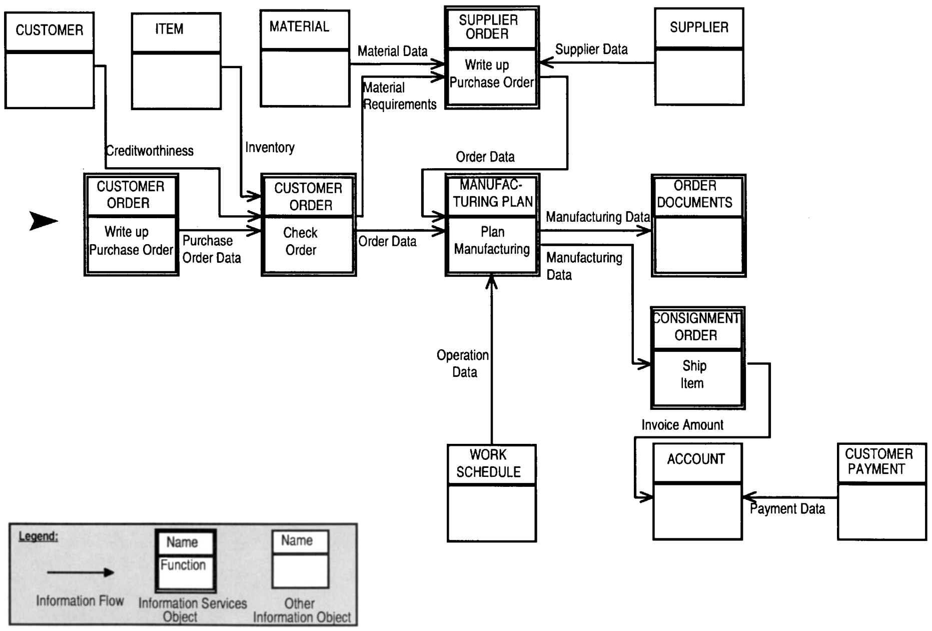


Figure 8 Information Flow of the Business Process "Order Processing."

mation services. For example, when orders are checked, the customer's credit is checked and inventory is checked for availability.

Because data flow is triggered by the functions that are linked to the information objects, it is more or less possible to read the function flow in Figure 8. However, if multiple functions are applied to an information object or multiple data flows are requested by a function, the function process cannot be uniquely deduced.

Besides information flow modeling, the (static) description of data structures is a very important modeling task. Static enterprise data models are used to develop proper data structures in order to implement a logically integrated database. Chen's entity relationship model (ERM) is the most widespread method for the conceptual modeling of data structures.

3.1.5. Process View

Building various views serves the purpose of structuring and streamlining business process modeling. Splitting up views has the added advantage of avoiding and controlling redundancies that can occur when objects in a process model are used more than once. For example, the same environmental data, events, or organizational units might be applied to several functions. View-specific modeling methods that have proven to be successful can also be used. Particularly in this light, view procedures differ from the more theoretical modeling concepts, where systems are divided into subsystems for the purpose of reducing complexity. In principle, however, every subsystem is depicted in the same way as the original system. This is why it is not possible to use various modeling methods in the same system.

It is important to note that none of the flows (organization, function, output, and information flow, respectively) illustrated above is capable of completely modeling the entire business process. We must therefore combine all these perspectives. To this end, one of the views should be selected as a foundation and then be integrated into the others. The function view is closest to the definition of a business process and is therefore typically used as a starting point. However, in the context of object-oriented enterprise modeling information, flows can serve as a starting point as well.

Figure 9 provides a detailed excerpt of our business process example, focusing the function "manufacture item" with all flows described above.

The method used to describe the process in Figure 9 is called event-driven process chain (EPC). The EPC method was developed at the Institute for Information Systems (IWi) of the Saarland

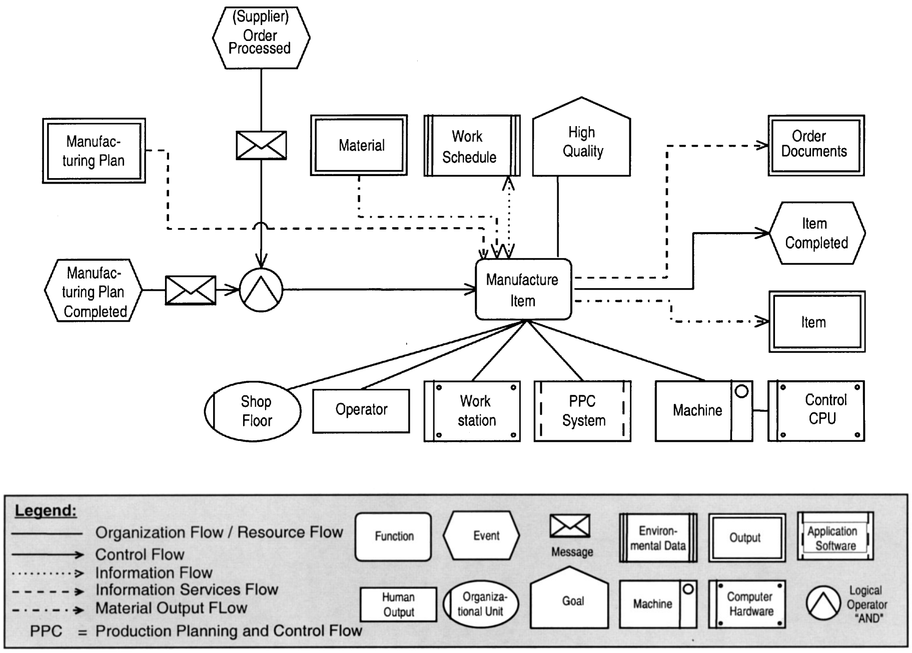


Figure 9 Detailed Excerpt of the Business Process "Order Processing."

University, Germany, in collaboration with SAP AG. It is the key component of SAP R/3's modeling concepts for business engineering and customizing. It is based on the concepts of stochastic networks and Petri nets. Simple versions exclude conditions and messages and include only E(vent)/A(ction) illustrations.

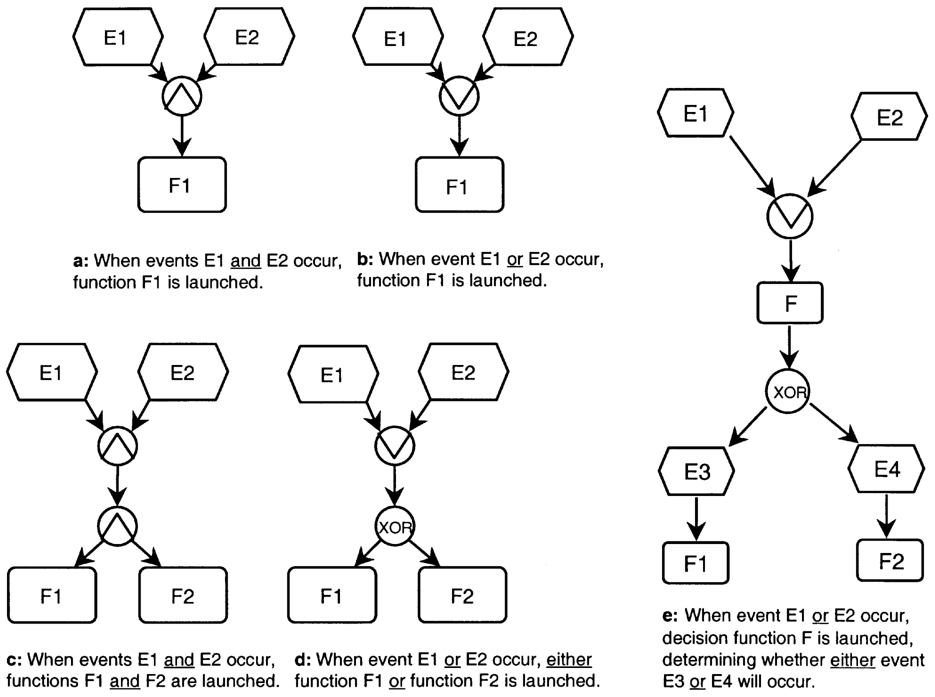
Multiple functions can result from an event. On the other hand, multiple functions sometimes need to be concluded before an event can be triggered. Logical relationships are illustrated by "and" ($\wedge$), "inclusive-or" ($\vee$) and "exclusive-or" (XOR) symbols. Figure 10 gives some typical examples of event relationships. When there are more complex relationships between completed functions and functions that have just been launched (such as different logical relationships between groups of functions), decision tables for incoming and outgoing functions, respectively, can be stored in an event.

3.2. Object-Oriented Enterprise Modeling

Object-oriented enterprise modeling starts with analyzing the entities of the real world. In the object-oriented model, these entities will be described through objects. The object data (attributes) represent characteristics of the real system and are accessible only by means of object methods. By their definition, attributes and methods objects are entirely determined. Objects that share the same characteristics are instances of the respective object class. Classes can be specialized to subclasses as well as generated to superclasses. This is called inheritance—each subclass will inherit the attributes and methods from its superclass.

We can equate the term *method* used in object-oriented analysis with the term *function*. Due to the fact that classes are frequently data classes (such as "customers," "suppliers," "orders," etc.), they represent the link between data and function view. We have already experienced object-oriented class design when we discussed the levels of abstraction and the examples of the modeling views (see Section 1.2 as well as Section 3.1), so we can skim the properties of creating object-oriented classes.

Object-oriented modeling is not based on a standardized method. Rather, a number of authors have developed similar or complementary approaches. The various graphical symbols they use for their approaches make comparisons difficult. The Unified Modeling Language (UML), introduced by



a: When events E1 <u>and</u> E2 occur, function F1 is launched.

b: When event E1 <u>or</u> E2 occur, function F1 is launched.

c: When events E1 <u>and</u> E2 occur, functions F1 <u>and</u> F2 are launched.

d: When event E1 <u>or</u> E2 occur, <u>either</u> function F1 <u>or</u> function F2 is launched.

e: When event E1 <u>or</u> E2 occur, decision function F is launched, determining whether <u>either</u> event E3 <u>or</u> E4 will occur.

Figure 10 Event Relationships in EPC.

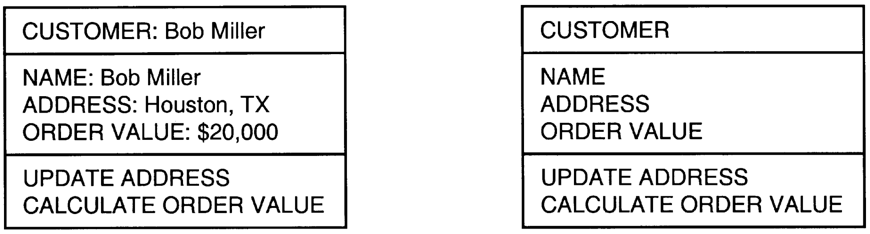


Figure 11 Object and Object Class.

Rumbaugh, Booch, and Jacobsen aims to streamline and integrate various object-oriented approaches, which is why we will use these symbols.

Objects, each with its own identity and indicated by an ID number, are described by properties (attributes). The functions (methods) that can be applied to the object define their behavior. Objects represent instances and are illustrated by rectangles. Objects with identical attributes, functionality, and semantics are grouped into object classes or regular classes. The quantity of customers thus forms the class "customer" (see Figure 11).

By naming attributes and methods, classes define the properties and the behavior of their instances, that is, objects. Because attributes and methods form a unit, classes realize the principle of encapsulation. In addition to attribute and method definitions for objects, we can also use class attributes and class methods that are valid only for the classes themselves, not for the objects. An example would be "number of customers" and "creating a new customer."

One significant property of the object-oriented approach is inheritance, giving classes access to the properties (attributes) and the behavior (methods) of other classes. Inherited attributes and methods can be overwritten and redefined by the inheriting class. Inheritance takes place within a class hierarchy with two types of classes, namely overriding classes and subordinate classes, as is shown by generalizing and specializing operations in data modeling. A class can also inherit properties from several overriding classes (multiple inheritance), with the resulting class diagram forming a network. Figure 12 gives an example of inheritance among object classes.

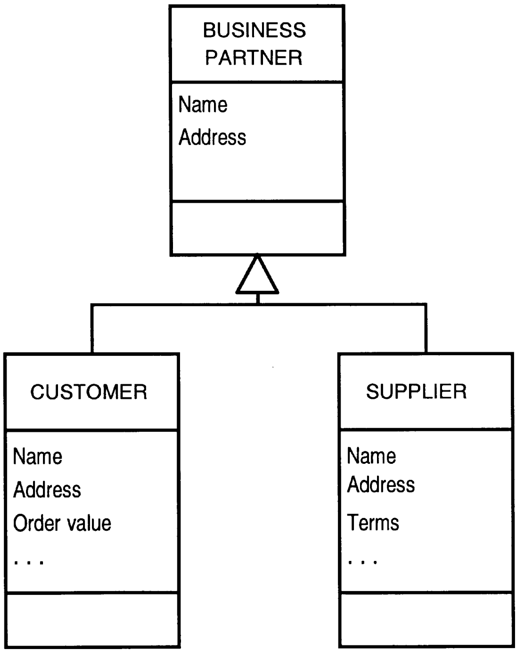


Figure 12 Inheritance.

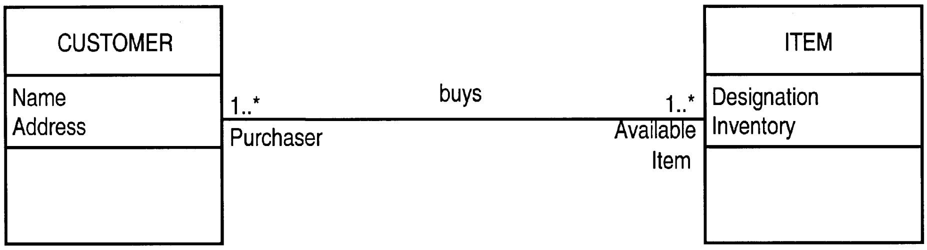


Figure 13 Association.

In addition to the generalizing relationships between classes, there are also relationships (i.e. associations) between objects of equal class ranking or between objects of the same class. These associations equate to relationships in entity relationship models, although here they are illustrated by only one line. These illustrations should be read from left to right. Cardinalities are allocated to an association. At each end of the association, role names can be allocated to the associations. If associations contain attributes, they are depicted as classes (see the class "purchasing process" in Figure 13).

Aggregations are a special kind of association describing the "part of" relationships between objects of two different classes. Role names can be applied to aggregations as well. If attributes are allocated to an aggregation, this leads to a class, as shown by the class "structure," in an aggregation relation for a bill of materials (see Figure 14).

Class definition, inheritance, associations, and aggregations make up the key structural properties of object-oriented modeling. Overall, the UML provides seven methods for describing the various static and dynamic aspects of enterprises. Nevertheless, even with methods such as use case diagrams and interaction diagrams, it is difficult to depict process branching, organizational aspects, and output flows. Therefore, one of the main disadvantages of the object-oriented approach is it does not illustrate business process in a very detailed manner.

4. MODELING ARCHITECTURES

4.1. Architecture of Integrated Information Systems (ARIS)

In the architecture of integrated information systems (ARIS), we can distinguish two different aspects of applications:

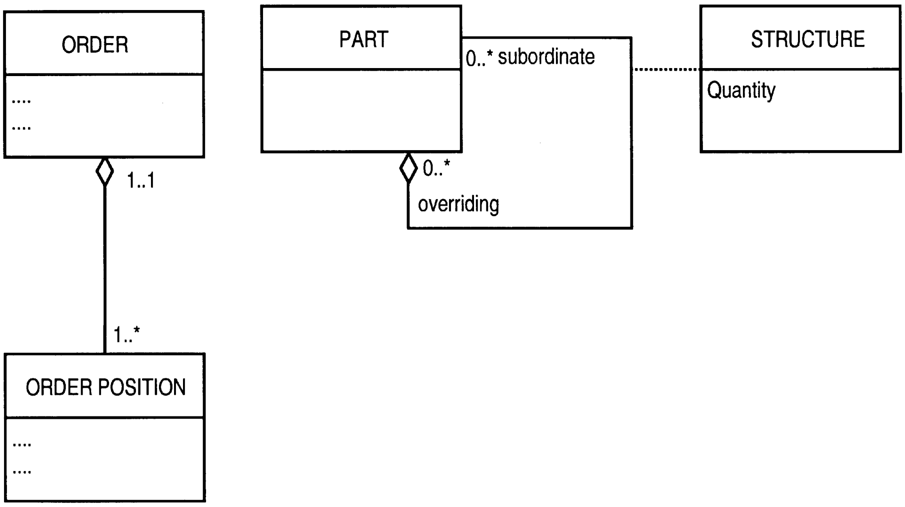


Figure 14 Aggregation.

1. The ARIS concept (ARIS house), an architecture for modeling enterprises, particularly describing business processes
2. The ARIS house of business engineering (HOBE), representing a concept for comprehensive computer-aided business process management.

Both applications are supported by the ARIS Toolset software system, developed by IDS Scheer AG. We will discuss tool support for enterprise modeling in Section 5.

The ARIS concept consists of five modeling views and a three-phase life cycle model. The integration of both leads to 15 building blocks for describing enterprise information systems. For each block the ARIS concept provides modeling methods, meta structures of which are included in the ARIS information model.

As we have already discussed static and dynamic modeling views (see Section 3.1), we can skim over the creation of modeling views. ARIS views are created according to the criterion of semantic correlation similarity, that is, enterprise objects that are semantically connected are treated in the same modeling view. The ARIS modeling views are (see Figure 15):

- *Data view:* This view includes the data processing environment as well as the messages triggering functions or being triggered by functions. Events such as "customer order received,"

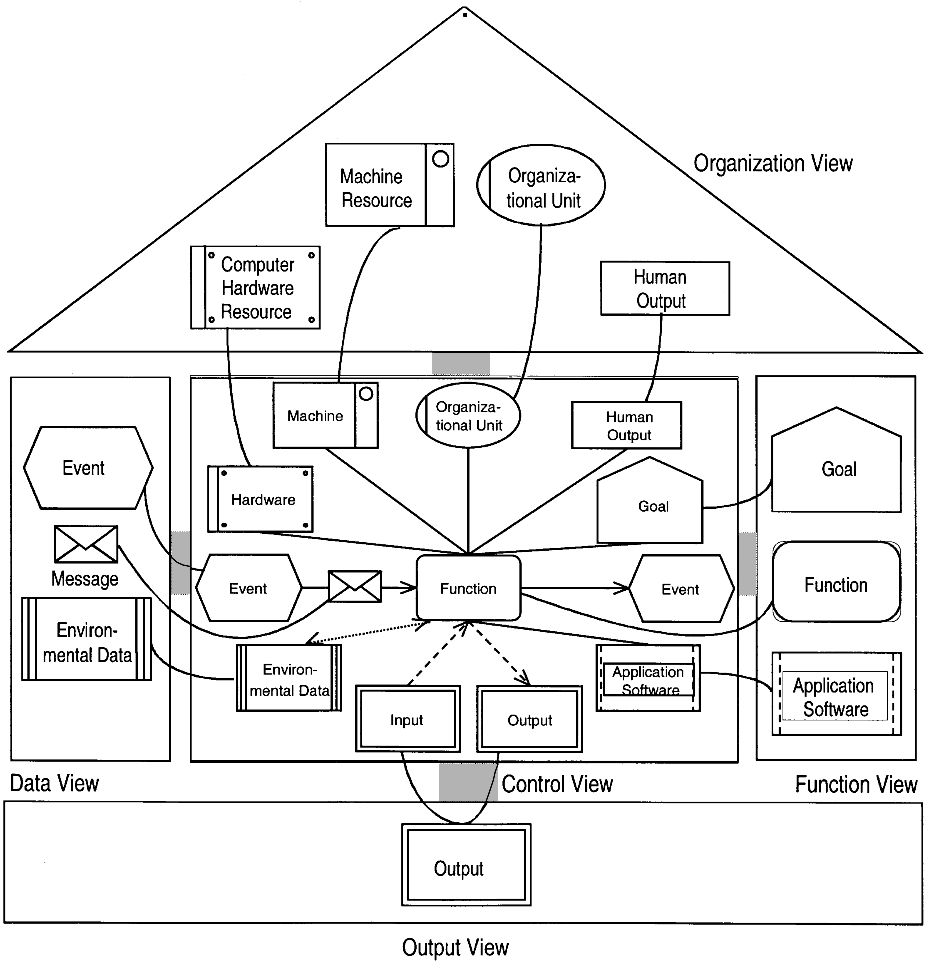


Figure 15 Views of the ARIS Concept.

"completion notice received," and "invoice written" are also information objects represented by data and therefore modeled in the data view.

- *Function view:* The business activities to be performed and their relationships form the function view. It contains the descriptions of each single activity (function) itself, the associations between super- and subordinate functions, and the logical sequence of functions. Enterprise goals that guide the performance of the functions and the application systems that support their execution are also components of this view.
- *Organization view:* Departments, office buildings, and factories, are examples of organizational units that perform business functions. They are modeled according to criteria such as "same function" and "same work object." Thus, the structure as well as the interaction between organizational units and the assigned resources (human resources, machine resources, computer hardware resource) are part of the organization view.
- *Output view:* This view contains all physical and nonphysical output that is created by business functions, including services, materials, and products as well as funds flows. Because each output is input for another function—even if this function is probably carried out by an external partner—input–output diagrams are also components of this view.
- *Control view/process view:* In the views described above, the classes are modeled with their relationships relative to the views. Relationships among the views as well as the entire business process are modeled in the control or process view. This enables a framework for the systematic inspection of all bilateral relationships of the views and the complete enterprise description using a process-oriented perspective.

The ARIS concept provides a set of methods that can be used to model the enterprise objects and their static and dynamic relationships. For each view at least one well-proven method is provided. Because ARIS is an open architecture, new modeling methods, such as UML methods, can easily be integrated into the meta structure. Figure 16 gives examples of ARIS methods for conceptual modeling.

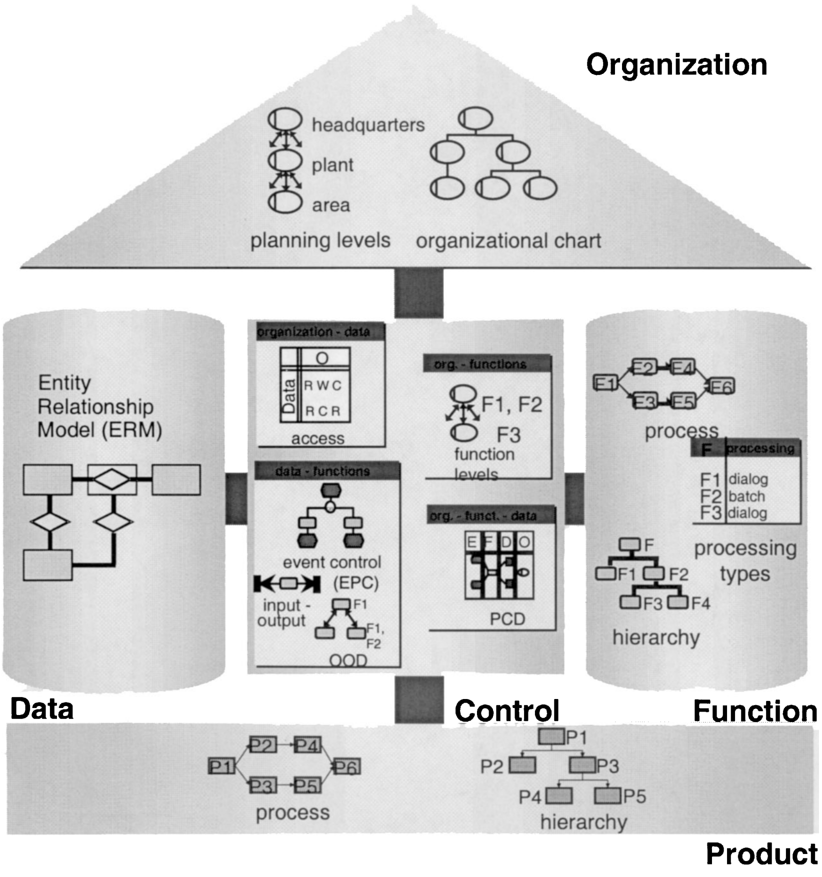
Up to now, we have discussed enterprises, particularly business processes, from a management point of view, that is, without any particular focus on information technology. The aforementioned application programs (components of the function view), computer hardware (a component of the organization view), and data media (components of the data view) contain only system names, not IT descriptions. These are included in the ARIS concept by evaluating the IT support provided by each ARIS view.

- Function views are supported by the application programs, which may be described in more detail by module concepts, transactions, or programming languages.
- Organization views, along with their production resources and the computer resources responsible, may be detailed further by listing network concepts, hardware components, and other technical specifications.
- Data views may be detailed more precisely by listing data models, access paths, and memory usage.
- Output views group the various types of output, such as material output and information services. Here again there is a close alignment with the supporting information technology. In material output (e.g., entertainment technology, automobiles, and machine tools), more and more IT components (e.g., chip technology), along with the necessary hardware, are used. Other service industries, such as airline reservations, are closely linked with IT as well.
- The fact that the respective views can be combined within the control view means that there is a definite link with IT, as demonstrated by the above arguments.

Using a phase model, enterprise models are thus transformed step by step into information and communication technology objects (see Figure 17).

The starting point of systems development is the creation of an IS-oriented initial strategic situation in Phase 1. "IS-oriented" means that basic IT effects on the new enterprise concepts are already taken into account. Some examples of these relationships might be creating virtual companies through communication networks, PC banking, integrated order processing and product development in industry (CIM), or integrated merchandise management systems (MMS) in retail.

Strategic corporate planning determines long-term corporate goals, general corporate activities, and resources. Thus, planning has an effect on the long-term definition of enterprises, influencing corporate goals, critical success factors, and resource allocation. The methods in question are similar to management concepts for strategic corporate planning. Provided actual business processes have already been described, this occurs in a general fashion. At this stage, it is not advisable to split up functions into ARIS views and then describe them in detail.



Legend:

ERM	=	Entity Relationship Model
Fn	=	Function n
E	=	Event
D	=	Data
O	=	Organizational Unit
Pn	=	Product n
R	=	Read
W	=	Write
C	=	Create
PCD	=	Process Chain Diagram
EPC	=	Event-driven Process Chain
OOD	=	Object Oriented Design

Figure 16 ARIS Methods for Conceptual Modeling.

In Phase 2, the requirements definition, individual views of the application system are modeled in detail. Here as well, business-organizational content is key. Examples of business processes should be included at this level. However, in this phase more conceptual modeling methods should be used than in the strategic approach because the descriptions for the requirements definition are the starting point for IT implementation. Modeling methods that are understandable from a business point of view should be used, yet they should be sufficiently conceptual to be a suitable starting point for a

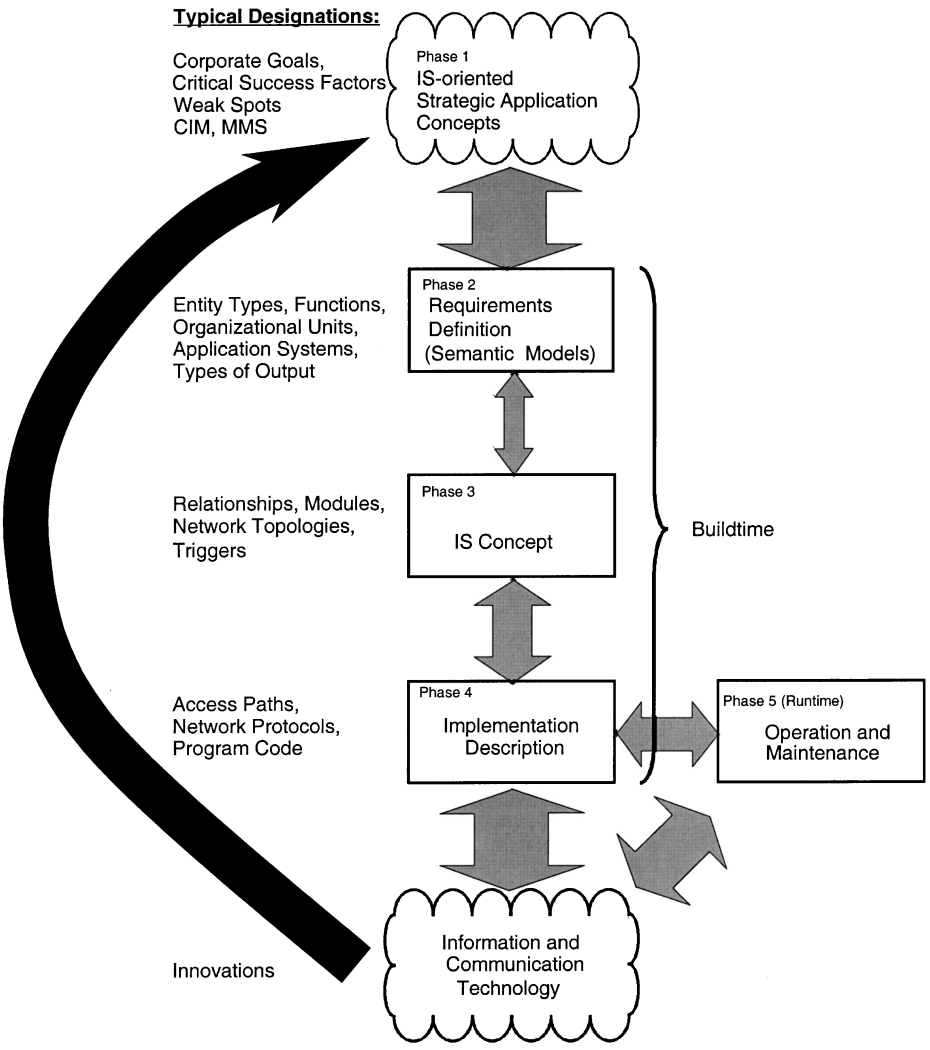

Typical Designations:
Corporate Goals, Critical Success Factors Weak Spots CIM, MMS
Phase 1
IS-oriented Strategic Application Concepts
Entity Types, Functions, Organizational Units, Application Systems, Types of Output
Phase 2
Requirements Definition (Semantic Models)
Relationships, Modules, Network Topologies, Triggers
Phase 3
IS Concept
Buildtime
Access Paths, Network Protocols, Program Code
Phase 4
Implementation Description
Phase 5 (Runtime)
Operation and Maintenance
Innovations
Information and Communication Technology

Figure 17 ARIS Phase Model.

consistent IT implementation. Therefore, it makes sense to include general IT objects, such as databases or programs, at this level.

Phase 3 calls for creating the design specification, where enterprise models are adapted to the requirements of the implementation tool interfaces (databases, network architectures, or programming languages, etc.). At this time, actual IT products are still irrelevant.

Phase 4 involves the implementation description, where the requirements are implemented in physical data structures, hardware components, and real-world IT products.

These phases describe the creation of an information system and are therefore known as buildtime. Subsequently, the completed system becomes operable, meaning it is followed by an operations phase, known as runtime. We will not address the operation of information systems, that is, runtime, in great detail.

The requirements definition is closely aligned with the strategic planning level, illustrated by the width of the arrow in Figure 17. However, it is generally independent of the implementation point of view, as depicted in the narrow arrow pointing to the design specification.

Implementation description and operations, on the other hand, are closely linked with the IT equipment and product level. Changes in the system's IT have an immediate effect on its type of implementation and operation.

The phase concept does not imply that there is a rigid sequence in the development process, as in the waterfall model. Rather, the phase concept also includes an evolutionary prototyping procedure. However, even in evolutionary software development, the following description levels are generally used. Phase models are primarily used because they offer a variety of description objects and methods.

The ARIS concept in Figure 18 is enhanced by the phases of the buildtime ARIS phase model. After a general conceptual design, the business processes are divided into ARIS views and documented and modeled from the requirements definition to the implementation description. These three description levels are created for controlling purposes as well. This makes it possible to create the links to the other components at each of the description levels.

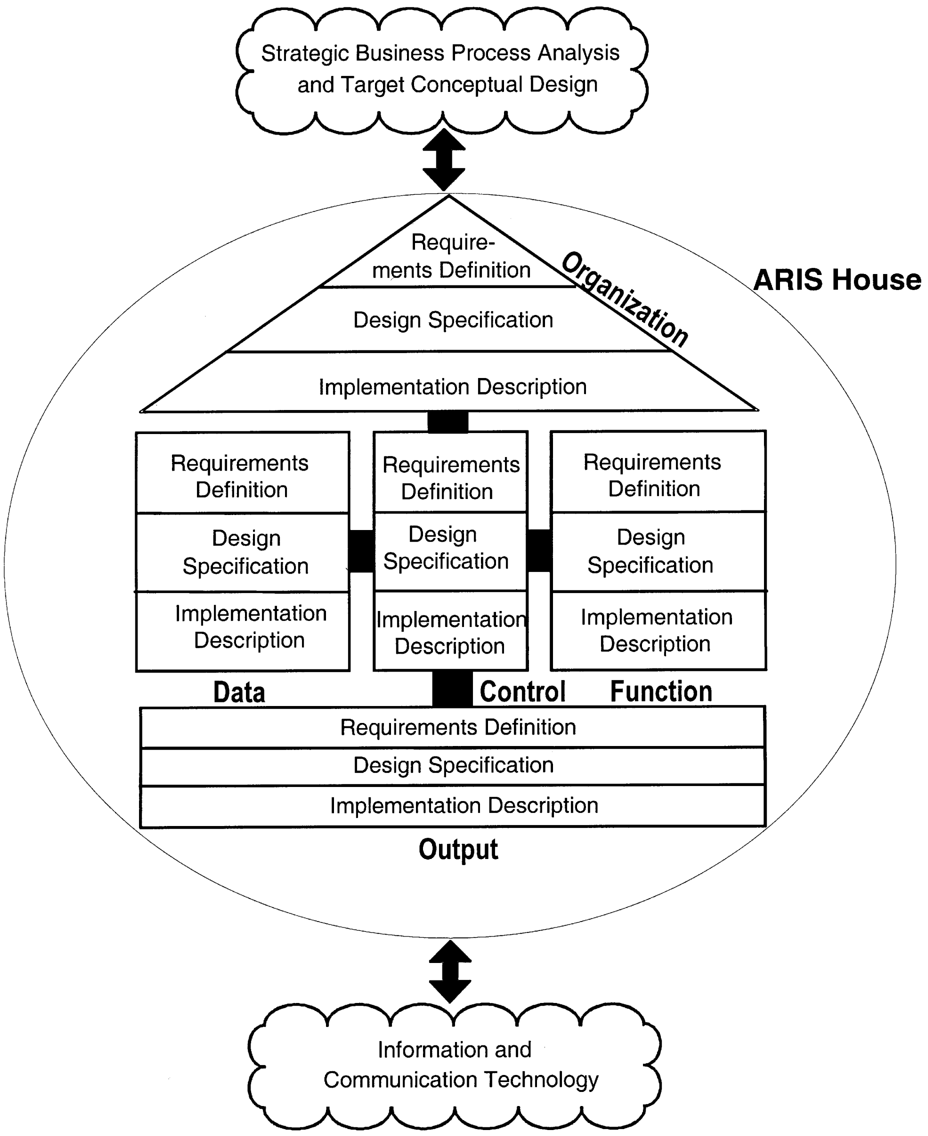


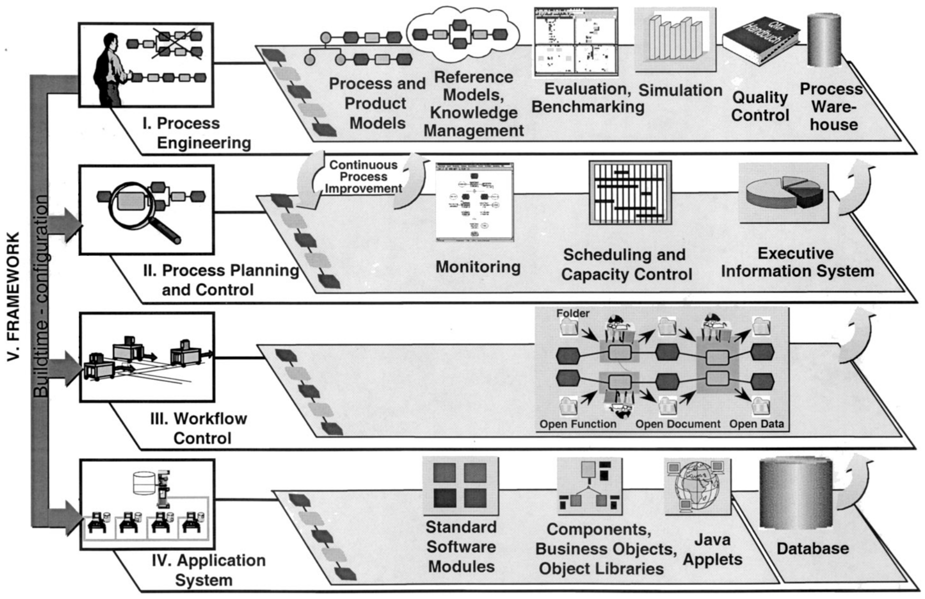
Figure 18 ARIS Concept with Phase Model.

The ARIS concept paves the way for engineering, planning, and controlling enterprises, particularly business processes. The ARIS house of business engineering (HOBE) enhances the ARIS concept by addressing comprehensive business process management from not only an organizational but an IT perspective. We will outline how ARIS supports business management in the design and development stages, using ARIS-compatible software tools.

Because business process owners need to focus on the ‘‘one-shot’’ engineering and description aspects of their business processes, ARIS HOBE provides a framework for managing business processes, from organizational engineering to real-world IT implementation, including continuous adaptive improvement. HOBE also lets business process owners continuously plan and control current business procedures and devote their attention to continuous process improvement (CPI). Comprehensive industry expertise in planning and controlling manufacturing processes is a fundamental component of HOBE. Objects such as ‘‘work schedule’’ and ‘‘bill of material’’ provide detailed description procedures for manufacturing processes, while production planning and control systems in HOBE deliver solutions for planning and controlling manufacturing processes. Many of these concepts and procedures can be generalized to provide a general process management system.

- At level I (process engineering), shown in Figure 19, business processes are modeled in accordance with a manufacturing work schedule. The ARIS concept provides a framework covering every business process aspect. Various methods for optimizing, evaluating, and ensuring the quality of the processes are also available.
- Level II (process planning and control) is where business process owners’ current business processes are planned and controlled, with methods for scheduling and capacity and (activity-based) cost analysis also available. Process monitoring lets process managers keep an eye on the states of the various processes.
- At Level III (workflow control), objects to be processed, such as customer orders with appropriate documents or insurance claims, are delivered from one workplace to the next. Electronically stored documents are delivered by workflow systems.
- At Level IV (application system), documents delivered to the workplaces are specifically processed, that is, functions of the business process are executed using computer-aided application systems (ranging from simple word processing systems to complex standard software solution modules), business objects, and Java applets.

HOBE’s four levels are linked with one another by feedback loops. Process control delivers information on the efficiency of current processes. This is where continuous adaptation and improve-



V. FRAMEWORK
Buildtime - configuration
I. Process Engineering
Process and Product Models
Reference Models, Knowledge Management
Evaluation, Benchmarking
Simulation
Quality Control
Process Ware-house
II. Process Planning and Control
Continuous Process Improvement
Monitoring
Scheduling and Capacity Control
Executive Information System
III. Workflow Control
Folder
Open Function
Open Document
Open Data
IV. Application System
Standard Software Modules
Components, Business Objects, Object Libraries
Java Applets
Database

Figure 19 ARIS House of Business Engineering.

ment of business processes in accordance with CPI begins. Workflow control reports actual data on the processes to be executed (amounts, times, organizational allocations) to the process control level. Application supporting modules are then started by the workflow system.

In the fifth component of the HOBE concept, Levels I through IV are consolidated into a framework. Frameworks contain information on the respective architecture and applications, configuring real-world applications from the tools at levels II and III as well as application information from the reference models (level I). Frameworks contain information on the composition of the components and their relationships.

4.2. Information Systems Methodology (IFIP-ISM)

Olle et al. (1991) give a comprehensive methodology for developing more traditional information systems. The designation ''methodology'' is used at the same level as ''architecture.'' The seven authors of the study are members of the International Federation for Information Processing (IFIP) task group, in particular of the design and evaluation of information systems working group WG 8.1 of information systems technical committee TC 8. The research results of the study are summarized in the guideline ''Information Systems Methodology.''

The design of the methodology does not focus on any particular IS development methods. Rather, it is based on a wide range of knowledge, including as many concepts as possible: IDA (interactive design approach), IEM (information engineering methodology), IML (inscribed high-level Petri nets), JSD (Jackson system development), NIAM (Nijssen's information analysis method), PSL/PSA (problem statement language/problem statement analyzer), SADT (structured analysis and design technique) as well as Yourdon's approach of object-oriented analysis.

This methodology is described by metamodels of an entity relationship concept. It features the point of view and stages of an information system life cycle, distinguishing data-oriented, process-oriented, and behavior-oriented perspectives (see Figure 20). Creating these perspectives is less a matter of analytical conclusion than simply of reflecting a goal of addressing the key issues typical in traditional IS developing methods.

Entity types and their attributes are reviewed in the data-oriented perspective. The process-oriented perspective describes events (business activities), including their predecessor or successor relationships. Events and their predecessor or successor relationships are analyzed in the behavior-oriented perspective.

From a comprehensive 12-step life-cycle model we will select three steps, information systems planning, business planning, and system design, and then examine the last two in detail in terms of their key role in the methodology.

Information systems planning refers to the strategic planning of an information system. In business analysis, existing information systems of an entire enterprise or of a subarea of the enterprise are

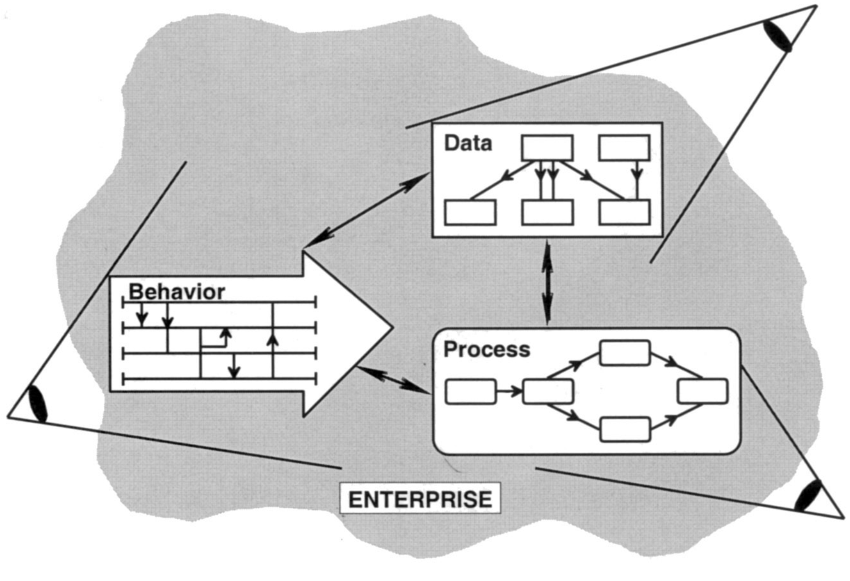


Figure 20 Perspectives of the IFIP Architecture. (From Olle et al. 1991, p. 13)

analyzed. The respective information system is designed in the step system design. This concept also includes a comprehensive procedural model, including a role concept for project organization.

With regard to ARIS, this concept has some overlapping areas. In others there are deviations. What both concepts have in common is their 2D point of view, with perspectives and development steps. There are differences in their instances, however. For example, Olle et al. do not explicitly list the organization view but rather review it along with other activities, albeit rudimentarily. The process definition more or less dovetails with ARIS's function definition. Data and functions or events and functions are also strictly separated from one another. The three perspectives linked together are only slightly comparable to ARIS control view. The step system design blends together the ARIS phases of requirements definition and design specification, with the emphasis on the latter.

4.3. CIM Open System Architecture (CIMOSA)

The ESPRIT program, funded by the European Union (EU), has resulted in a series of research projects for developing an architecture, Computer Integrated Manufacturing Open System Architecture (CIMOSA), for CIM systems. CIMOSA results have been published by several authors, including Vernadat (1996). This project originally involved 30 participating organizations, including manufacturers as the actual users, IT vendors, and research institutes. Although the project focused on CIM as an application goal, its mission was to provide results for general enterprise modeling. One of CIMOSA's goals was also to provide an architecture and a methodology for vendor-independent, standardized CIM modules to be "plugged" together, creating a customer-oriented system ("plug and play").

The CIMOSA modeling framework is based on the CIMOSA cube (see Figure 21).

CIMOSA distinguishes three different dimensions, described by the three axes of the cube. The vertical direction (stepwise derivation) describes the three description levels of the phase concept:

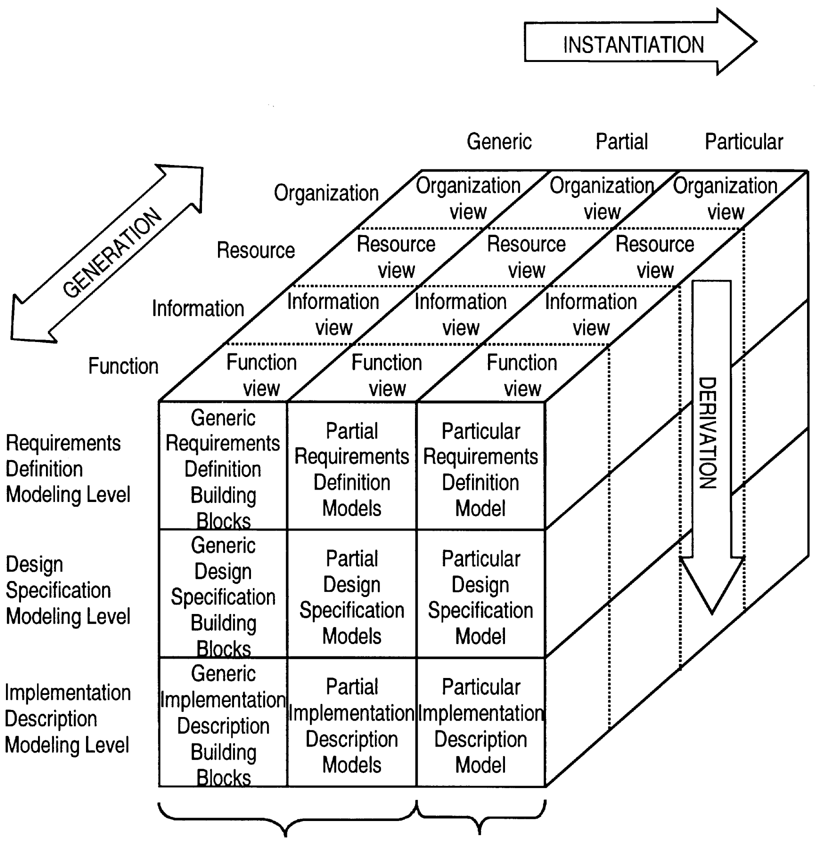

INSTANTIATION
Generic
Partial
Particular
GENERATION
Organization
Resource
Information
Function
Organization view
Resource view
Information view
Function view
DERIVATION
Requirements Definition Modeling Level
Design Specification Modeling Level
Implementation Description Modeling Level
Generic Requirements Definition Building Blocks
Partial Requirements Definition Models
Particular Requirements Definition Model
Generic Design Specification Building Blocks
Partial Design Specification Models
Particular Design Specification Model
Generic Implementation Description Building Blocks
Partial Implementation Description Models
Particular Implementation Description Model

Figure 21 The CIMOSA Modeling Architecture. (CIMOSA cube). (From Vernadat 1996, p. 45)

requirements definition, design specification, and implementation description. These levels are for the most part identical with those of the ARIS life cycle.

In the horizontal dimension (stepwise instantiation), concepts are individualized step by step. First, basic requirements (generic requirements, building blocks) are defined, then particularized in the next step according to industry specific requirements (partial requirements). In Step 3, they are broken up into enterprise-specific requirements (particular requirements).

This point of view makes it clear that initially, according to CIMOSA, general building blocks should be used to define standards, after which the building blocks are grouped into industry specific reference models. In the last step, they are used for developing enterprise-specific solutions. In ARIS, the degree of detailing an information model is defined while the granularity issues are addressed.

By directly entering content-related reference models, the CIMOSA architecture, it becomes clear, combines general methodological issues regarding information systems and application-related CIM domains.

The third dimension, stepwise generation, describes the various views of an information system. This point of view has goals similar to ARIS regarding the creation of views, although not all the results are the same. CIMOSA divides description views into function view, information view, resource view, and organization view. Function view is the description of events, although it also includes a combination of other elements such as events and processes, including performance and exception handling. Information view refers to the data view or object definition. Resource view describes IT and production resources, and organization view implies the hierarchical organization.

CIMOSA also breaks up the entire context into various views, although it lacks a level for reassembling them, as opposed to ARIS with its control and process views. This results in the fact that in CIMOSA, descriptions of the individual views are combined with one another. For example, when resources are being described, they are at the same time also allocated to functions. The CIMOSA modeling concept does not feature an output view.

The CIMOSA concept develops an architecture suitable for describing information systems, into which content in the form of standardized reference models, all the way to actual software generation, can be entered. Despite the above-mentioned drawbacks, it considers important points. Based on this concept, modeling methods are classified in CIMOSA and described by metamodels, all the while adhering to an event-driven, business process-oriented view. Furthermore, enterprises are regarded as a series of multiple agents communicating with one another.

Despite the considerable financial and intellectual efforts spent on CIMOSA, its practical contribution so far has been minimal. Business users involved in the project have so far reported few special applications resulting therefrom, with the exception of the car manufacturer Renault with a repair service application for manufacturing plants and the tooling company Traub AG with an application for optimizing individual development of tools. To date, a CIMOSA-based modeling tool has not been used much in practice.

The main reason for the lack of success in real-world applications is presumably its very theoretical design, which does not incorporate commercially available IT solutions (standard software, for example). Considering the general lack of interest in CIM concepts, the extremely specialized focus of this approach seems to be working to its disadvantage.

4.4. Zachman Framework

A framework for describing enterprises, quite popular in the United States, was developed by J. A. Zachman. This concept is based on IBM's information systems architecture (ISA) but has been enhanced and presented by Zachman in numerous talks and seminars.

This approach (see Figure 22) consists of six perspectives and six description boxes. In the ARIS terminology, Zachman's description boxes would equate to views and perspectives would equate to the levels of the life-cycle model.

Perspectives are listed in brackets along with the respective role designations of the party involved: scope (planner), enterprise model (owner), system model (designer), technology model (builder), components (subcontractor), and functioning system (user).

The areas to be viewed are designated by interrogatives with the respective actions listed in brackets: what (data), how (function), where (network), who (people), when (time), and why (rationale). Perspectives and files to be viewed are at a right angle to one another. Every field in the matrix is described by a method.

Contrary to ARIS, the Zachman framework is not capable of being directly implemented into an information system and the relationships between the description fields are not entered systematically. Furthermore, the relationship of Zachman's framework with the specific creation of output within the business process is not apparent.

Initial approaches for supporting tools are becoming apparent, thanks to cooperation with Framework Software, Inc.

FOCUS

	Generic Framework Element Bond Element	WHAT *(Dato)* Entity Relationship Entity	HOW *(Function)* Process Input-Output Process	WHERE *(Network)* Node Line Node	WHO *(People)* Agent Work Agent	WHEN *(Time)* Event Cycle Event	WHY *(Rationale)* End Means End
PERSPECTIVE	SCOPE *(Planner)*	Entity List	Process List	Location List	Organization List	Major Event List	Objective List
	ENTERPRISE MODEL *(Owner)*	Enterprise Entity Enterprise Rule Enterprise Entity	Enterprise Process Resource Enterprise Process	Enterprise Location Enterprise Channel Enterprise Location	Organization Work Organization	Enterprise Event Enterprise Cycle Enterprise Event	Objective Strategy Objective
	SYSTEM MODEL *(Designer)*	Entity Type Relationship Type Entity Type	System Process User View System Process	Site Link Site	Role Presentation Role	System Event System Cycle System Event	Criterion Choice Criterion
	TECHNOLOGY MODEL *(Builder)*	Data Structure Referential Integrity Data Structure	Application Device Format Application	Connection Point Communication Line Connection Point	User Technical Interface User	Technical Event Technical Cycle Technical Event	Condition Action Condition
	COMPONENTS *(Sub-contractor)*	Data Container Aquisition Data Container	Module/Object Couple/Message Module/Object	Address Protocol Address	Individual Transaction Individual	Component Event Component Cycle Component Event	Sub-condition Step/Task Sub-condition
	FUNCTIONING SYSTEM *(User)*	Information Integrity Information	Procedure Request Procedure	Client/Server Access Client/Server	Worker Work Session Worker	Operating Event Operating Cycle Operating Event	Target Option Target

Figure 22 Zachman Framework. (From Burgess and Hokel 1994, p. 26, © Framework Software, Inc. All rights reserved)

5. MODELING TOOLS

5.1. Benefits of Computerized Enterprise Modeling

Modeling tools are computerized instruments used to support the application of modeling methods. Support in designing enterprise models through the use of computerized tools can play a crucial role in increasing the efficiency of the development process. The general benefits of computer support are:

- All relevant information is entered in a structured, easy-to-analyze form, ensuring uniform, complete documentation. The incorporation of a user-friendly graphical interface offers a comfortable means of creating and modifying the necessary diagrams. Changes in one part of the documentation are automatically updated in other locations and thus need only be performed once.
- Referencing stored design to its creation date enables the different versions to be managed in a comfortable fashion. Versions are overwritten only if the user expressly wishes it; otherwise, each modification is maintained as an independent version.
- The use of a tool facilitates conformance to a fixed procedural model. Consistency checks ensure that modeling activities are not permitted until the preceding level has been completely processed.
- The internal consistency of the draft model can be validated by using rule-based systems. The general system requirements must also be entered within the framework of tool application in order to ensure external consistency. This necessitates the use of a powerful tool that provides structured support for the entire life cycle.
- Generation algorithms enable graphical representations to be derived from existing database structures. Likewise, the graphical tool interface enables the information objects to be categorized in a meaningful, content-related manner. However, since the aim of providing application-oriented data structuring is again based on content-related perspectives, it would be difficult for a tool to provide further support. In particular, reclassifying basic information units such as attributes into information objects that are more meaningful from a subject-oriented perspective is the virtual equivalent of redesigning the model.

5.2. Characterization of Modeling Tools

We can distinguish different groups of tools for enterprise modeling (see Figure 23). In relation to the different tasks of enterprise modeling, each group shows its special support profile.

	Programming Environments	CASE Tools	Drawing Tools	BPI Frameworks	ERP Software
Documentation of Processes		•	⬤	⬤	●
Analysis of Processes		•	●	⬤	•
Requirements Definition		•	•	⬤	●
Design Specification	•	⬤		●	⬤
Implementation Description	⬤	●		•	⬤

Figure 23 Modeling Tools.

The first group that can be seen as modeling tools in a broader sense are programming environments. Programming environments such as Borland JBuilder and Symantec Visual Café for Java clearly emphasize the phase of implementation description. They describe a business process by the language of the software application that is used to support the process performance. In few cases, programming languages provide limited features for design specification.

CASE tools such as ADW, on the other hand, support a specific design methodology by offering data models and function models (e.g., the entity relationship method (ERM) and structured analysis and design technique [SADT]). Their metastructure has to reflect this methodology. Thus, they automatically provide an information model for the repository functionality of this system. Because CASE tools cannot really claim to offer a complete repository, they are also called encyclopedias.

In contrast, drawing tools such as VISIO and ABC FlowCharter support the phases of documentation and, to a certain extent, analysis of business structures and processes. The emphasis is on the graphical representation of enterprises in order to better understand their structures and behavior. In most of the cases, drawing tools do not provide an integrated meta model, that is, repository. Consequently, drawing tools cannot provide database features such as animating and simulating process models, analyzing process times, and calculating process costs.

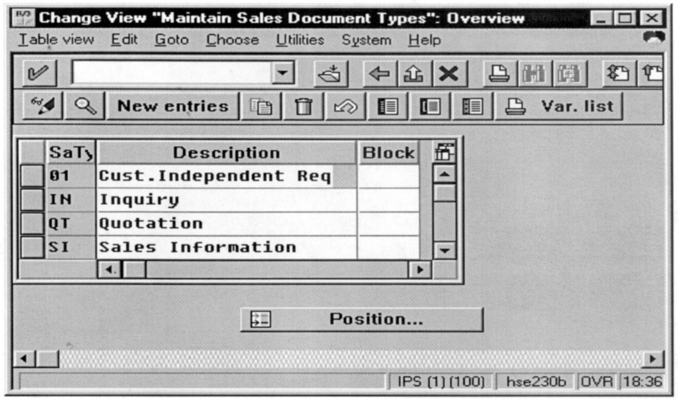
Business process improvement (BPI) frameworks such as ARIS offer a broader set of modeling methods, which are described together in one consistent metamodel. As a result, each model is stored in the framework's repository and thus can be retrieved, analyzed, and manipulated and the model history can be administered (see Section 4.1).

The last group encompasses standard software systems, such as enterprise resource planning (ERP) systems like those from SAP, Baan, or Oracle. ERP systems offer greater flexibility with respect to customization, provide support for data management, and include functionality for creating and managing a (sub)repository. The main weakness is in active analysis of business structures—that is, ERP tools typically do not offer tools for simulation process alternatives.

Because all the modeling tools presented focus on different aspects, none of the systems is suitable for handling the entire systems development process. However, ERP systems are opening up more and more to the modeling and analysis level, endeavoring to support the whole system life cycle.

The integration between ARIS and SAP R/3, for example, demonstrates seamless tool support from the documentation to the implementation phase. In an excerpt from the SAP R/3 reference model, Figure 24 shows a customizing example with the ARIS Toolset. For clarification purposes, the four windows are shown separately. In a real-world application, these windows would be displayed on one screen, providing the user with all the information at once.

The upper-right window shows an excerpt from the business process model in the ARIS modeling tool, illustrating the part of the standard software process that can be hidden. An additional process branch, not contained in the standard software, must be added.



Change View "Maintain Sales Document Types": Overview

Table view Edit Goto Choose Utilities System Help

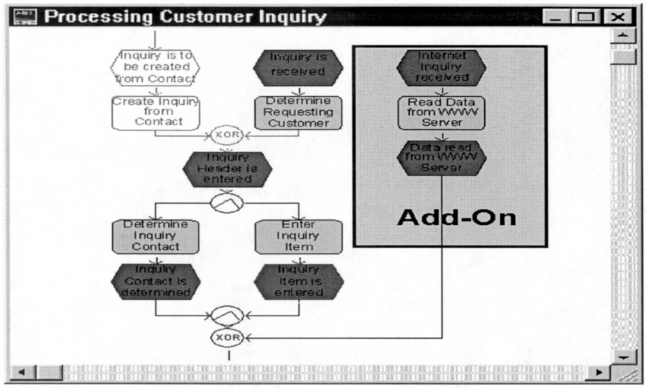
New entries | Var. list

SaTy	Description	Block
01	Cust.Independent Req	
IN	Inquiry	
QT	Quotation	
SI	Sales Information	

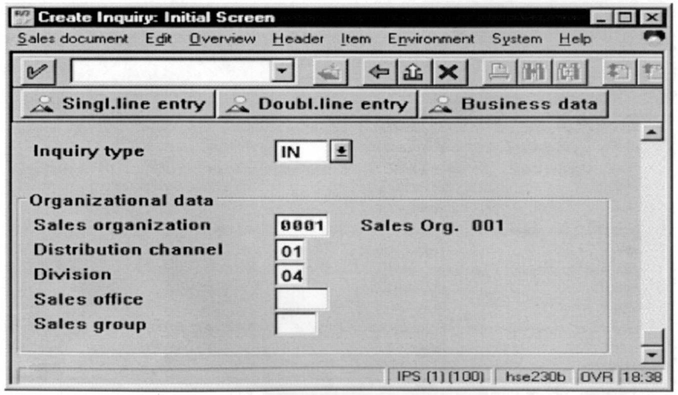
Position...

IPS (1) (100) | hse230b | OVR | 18:36

Customizing: Maintain Sales Document Types


Processing Customer Inquiry
Inquiry is to be created from Contact
Inquiry is received
Create Inquiry from Contact
Determine Requesting Customer
XOR
Inquiry Header is entered
Determine Inquiry Contact
Enter Inquiry Item
Inquiry Contact is determined
Inquiry Item is entered
XOR
Internet Inquiry received
Read Data from WWW Server
Data read from WWW Server
Add-On

Process Model



Create Inquiry: Initial Screen

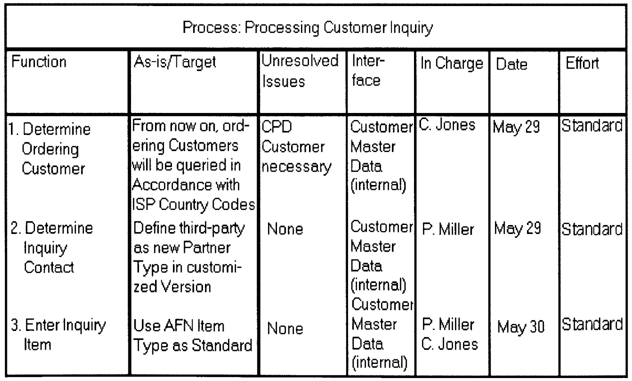
Sales document Edit Overview Header Item Environment System Help

Singl.line entry | Doubl.line entry | Business data

Inquiry type	IN
Organizational data	
Sales organization	0001 Sales Org. 001
Distribution channel	01
Division	04
Sales office	
Sales group	

IPS (1) (100) | hse230b | OVR | 18:38

Function „Create Inquiry"



Process: Processing Customer Inquiry

Function	As-is/Target	Unresolved Issues	Inter-face	In Charge	Date	Effort
1. Determine Ordering Customer	From now on, ord-ering Customers will be queried in Accordance with ISP Country Codes	CPD Customer necessary	Customer Master Data (internal)	C. Jones	May 29	Standard
2. Determine Inquiry Contact	Define third-party as new Partner Type in customi-zed Version	None	Customer Master Data (internal)	P. Miller	May 29	Standard
3. Enter Inquiry Item	Use AFN Item Type as Standard	None	Customer Master Data (internal)	P. Miller C. Jones	May 30	Standard

Documentation of Results

Figure 24 Customizing SAP R/3 with ARIS Toolset.

The function "create inquiry" asks users which screen is being used in SAP R/3. Using the process model as a modeling tool and by clicking on the function (or starting a command), users can seamlessly invoke SAP R/3. This screen is shown at the bottom left.

The Implementation Management Guide (IMG) customizing tool is activated for customizing the function. In the upper-left hand window, it depicts the function parameters that are available.

With the modeling tool, results of discussions, parameter decisions, unresolved issues, and the like are stored in the function, as depicted in the bottom-right window. This enables detailed documentation of business and IT-specific business process engineering. This documentation can be used at a later point in time for clarifying questions, using this knowhow for follow-up projects, and monitoring the project.

6. MODELING OUTLOOK

In the mid-1990s, the BPR debate drew our attention from isolated business activities to entire value chains. Yet "entire" process management in most of the cases focused on the information flow within departmental, corporate, or national boundaries. Obstacles within those areas appeared hard enough to cope with. Therefore, interorganizational communication and cooperation were seldom seriously put on the improvement agenda. Consequently, enterprise models, particularly business process models were also restricted to interorganizational aspects.

In the meantime, after various BPR and ERP lessons learned, companies seem to be better prepared for business scope redefinition. More and more, they sense the limitations of interorganizational improvement and feel the urge to play an active role in the global e-business community. That means not only creating a company's website but also designing the back-office processes according to the new requirements.

Obviously, this attitude has effects on business application systems, too. While companies are on their way to new business dimensions, implemented ERP systems cannot remain inside organizational boundaries. On the technical side, ERP vendors are, like many other software vendors, forced to move from a traditional client–server to a browser–web server architecture in order to deliver e-business capabilities. Hence, for their first-generation e-business solutions almost all big ERP vendors are using a mixed Java/XML strategy. On the conceptual side, ERP vendors are facing the even bigger challenge of providing instruments for coping with the increasing e-business complexity. Business process models appear to be particularly useful in this context. While e-business process models are fundamentally the same as interorganizational process models, integration and coordination mechanisms become even more important:

- Due to increasing globalization, e-business almost inevitably means international, sometimes even intercultural, business cooperation. While ERP systems were multilingual from the very first, the human understanding of foreign business terms, process-management concepts, legal restrictions, and cultural individualities is much more difficult. Because models consist of graphic symbols that can be used according to formal or semiformal grammars, they represent a medium that offers the means to reduce or even overcome those problems.
- Many improvement plans fail because of insufficient transparent business processes and structures. If people do not realize the reengineering needs and benefits, they will not take part in a BPR project and accept the proposed or made changes. While this is already a serious problem within organizational boundaries, it becomes even worse in the case of interorganizational, that is, geographically distributed, cooperation. In the case of business mergers or virtual organizations, for example, the processes of the partners are seldom well known. Yet, in order to establish a successful partnership, the business processes have to be designed at a very high level of detail. Thus, business process models can help to define the goals, process interfaces, and organizational responsibilities of interorganizational cooperation clearly.
- Up to now, we have mainly discussed strategic benefits of modeling. However, at an operational level of e-business, models are very useful, too. In business-to-business applications such as supply chain management we have to connect the application systems of all business partners. This means, for example, that we have to fight not only with the thousands of parameters of one single ERP system but with twice as many, or even more. Another likely scenario is that the business partners involved in an interorganizational supply chain are using software systems of different vendors. In this case, a business process model can first be used to define the conceptual value chain. Secondly, the conceptual model, which is independent from a certain software, can be connected to the repositories of the different systems in order to adopt an integrated software solution.

These few examples already demonstrate that enterprise models play a major role in the success of e-business. Some software vendors, such as SAP and Oracle, have understood this development and are already implementing their first model-based e-business applications.

REFERENCES

Burgess, B. H., and Hokel, T. A. (1994), *A Brief Introduction to the Zachman Framework,* Framework Software.

Olle, T. W. et al. (1991), *Information System Methodologies: A Framework for Understanding,* 2nd ed., Addison-Wesley, Wokingham.

Vernadat, F. B. (1996), *Enterprise Modeling and Integration: Principles and Applications,* Chapman & Hall, London.

ADDITIONAL READING

Scheer, A.-W., *Business Process Engineering: Reference Models for Industrial Enterprises,* 2nd ed., Springer, Berlin, 1994.

Scheer, A.-W., *ARIS—Business Process Frameworks,* 2nd Ed., Springer, Berlin, 1998.

Scheer, A.-W., *ARIS—Business Process Modeling,* 2nd Ed., Springer, Berlin, 1999.