Building business and application systems with the Retail Application Architecture

by P. Stecher

An industry application architecture is a framework for integrating applications and databases and can also be used for analyzing and re-engineering the business of an enterprise as a whole, provided it is structured correctly. This paper describes the motivation, structure, and possible uses of the Retail Application Architecture™ (RAA™). The core of RAA is a set of generic enterprise models for companies in the retail and wholesale distribution industry. RAA is oriented as much to the business expert as to the information systems (I/S) department. The goal of RAA is to contribute to the task of building sound business systems in a more efficient and effective manner.

When we look at the current situation of computerized applications and information in many enterprises, we witness the existence of a variety of applications that often are isolated, sometimes overlapping, but in essence unrelated. Both the information systems (I/S) and user departments have contributed to this state of affairs: the I/S department by developing or buying programs and applications at different times with little concern for how they interface; and the business users, impatient with the delivery of applications from the I/S department, by embarking on the development of their own solutions, often on small computers controlled by them. Not only were these applications often unrelated, but this situation resulted in an uncontrolled proliferation of data. Rather than seeing shared data as a company asset from which consistent decisions can be made, we encounter multiple, sometimes contradictory, versions of the same information, thus giving rise to wrong decisions.

To solve these problems, an approach is needed to integrate applications and data. This approach must be in a language that is comprehensible to the business users and thus ensure their full participation in the creation of integrated systems. It must facilitate the management of data as a company asset. The Retail Application Architecture* (RAA*) is intended to be such an approach for the retail and wholesale industries.

Initially application architecture was being used in support of the I/S strategy and application development. With RAA we advocate other uses such as application positioning, business analysis, and business re-engineering. These uses are geared to offer competitive business advantages in a company. In more and more industries, the most advanced companies succeed by reviewing and re-engineering their business systems, i.e., the way in which they do business inside their companies and with customers and suppliers. The challenge in building a retail and wholesale

distribution industry application architecture is to define the architecture in such a way that it can be applied to individual retail systems across various types of retailing, e.g., hypermarket, supermarket, or specialty store.

RAA edition 1 has been available as a service offering by IBM since January 1992.[1] It can provide business analysis, definition of an information technology and application strategy, application development, and database development.

In the first section of this paper, we review various origins and schools of thought about application architecture. We then define the RAA components and give the rationale for their structure. The subsequent sections deal with the use of RAA and address questions such as: "How can RAA, which was devised for a whole industry, be applied to a particular retail enterprise?," "How can RAA be employed to analyze and re-engineer a retail business system?," "What role can RAA play in the field of applications and decision support systems?"

What is application architecture?

In this section we give a historical perspective of application architecture and the role enterprise modeling plays in it. This perspective allows us to better position RAA.

Application architecture for a particular company. Application architecture is an umbrella term (or we might say, another "buzzword") for various concepts and constructs. Among them we find: enterprise modeling, business modeling, data modeling, and application development methodology. In the construction of buildings, architecture refers to the style and method of design and construction. Similarly, application architecture acts as a blueprint guiding the construction of applications. Application architecture emerged when applications in a particular company became diverse over time and went in different directions. It was recognized that some kind of high-level application design was required. This high-level design had to define processes, data, and the locations where processes and data resided. The representational techniques were those being used for application design in general, for instance, decomposition diagrams, flow diagrams, and relational charts. The high-level design was eventually called application architecture. Application architecture was introduced at this particular company as a frame to bring their applications together again. It was soon realized that application architecture could also provide a common base for applications in the early stage of their life cycle; by modeling a particular business system, a point of reference was created when applications in support of the business system were finally built. Architecture differs from design in that it is geared to a long-term goal, represents a broad perspective, and addresses requirements rather than a specific solution.

Figure 1 illustrates the evolution of applications from a time when they are developed and used in unique, diverse ways to a time when their development is done in accordance with an architecture.

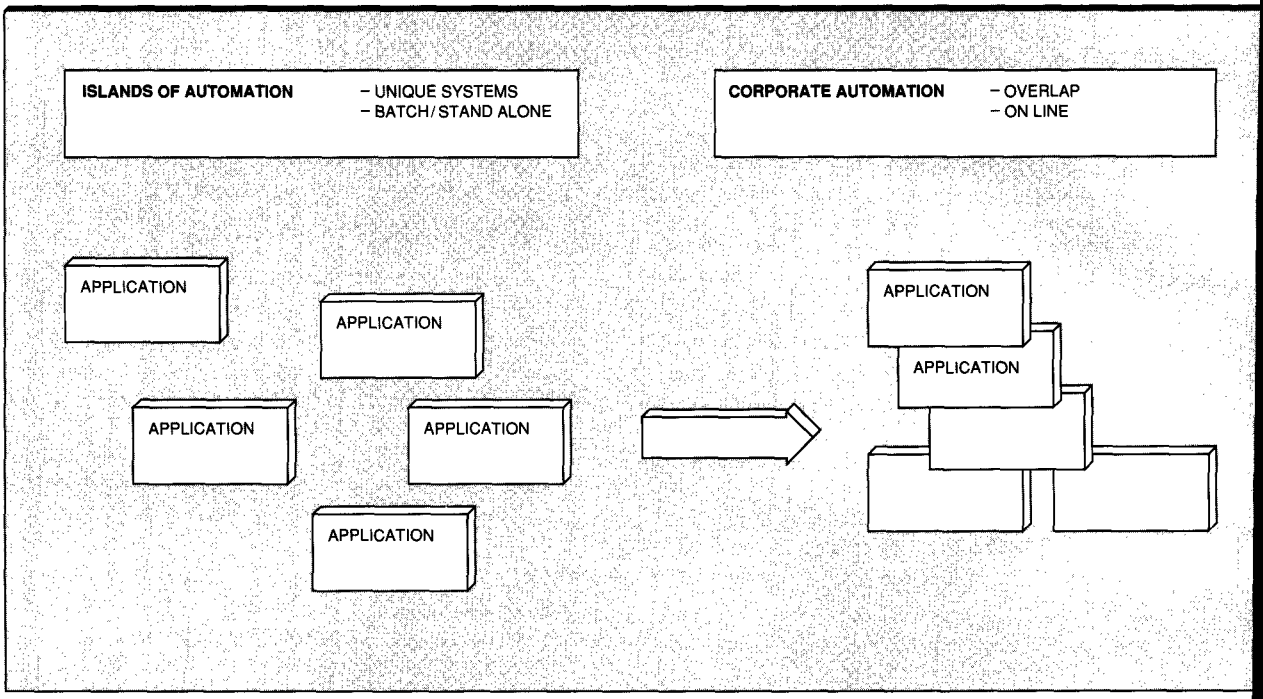
Whichever purpose application architecture served, it was initially for a particular company.

From hardware architecture to industry application architecture. There is, however, another origin of application architecture. This one does not concern itself with a particular phenomenon, but with a range of phenomena, all belonging to a class. The features of this class are specified by an architecture.

In the beginning, the term architecture was used in information technology (I/T) solely to define hardware specifications. When IBM introduced the System/360* in 1964, the most remarkable feature was the fact that the same architecture was implemented from the smallest to the largest processors. In 1974 IBM announced Systems Network Architecture (SNA). SNA has a hardware part that permits the interconnection of communication nodes and terminals, but it also has a software part that brought the capability to more easily run and manage extremely large networks. SNA is a good example of an architecture that was announced at an early stage, was filled in by hardware and software products over many years, and is still evolving as technology matures.

In the meantime we have seen the use of the term architecture spreading to applications. In 1987 IBM introduced Systems Application Architecture* (SAA*) as its strategic framework for developing and porting applications across diverse hardware architectures.[2] SAA has been designed to be an open, responsive architecture that will

Figure 1 Applications evolution



continue to evolve as customer requirements change and new technology becomes available.

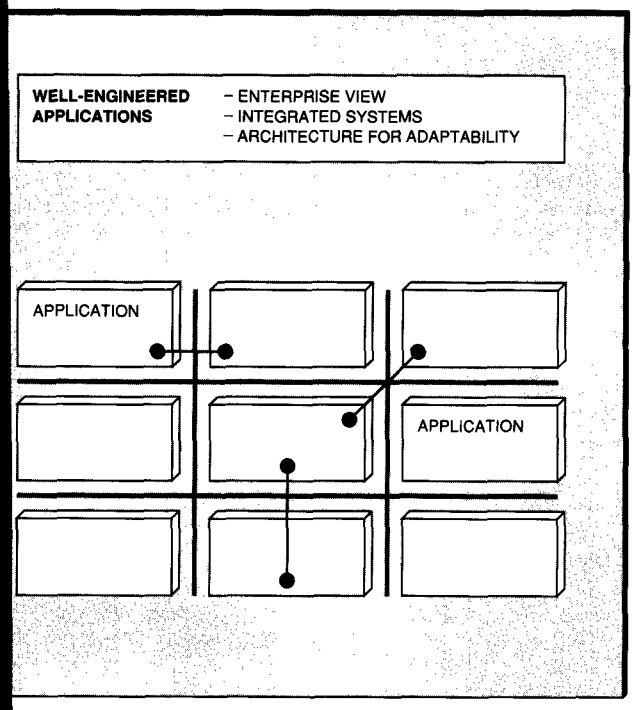
Both SAA and RAA are software architectures. How do they compare? SAA is technology oriented. RAA, in comparison, is business oriented. Emphasis is on how applications help retailers solve problems. RAA defines an overall structure for applications to support information sharing and business process integration across a retail company (Figure 2).

An industry application architecture is intended to develop an enterprise-wide perspective of business activities and information to be used by these activities. An enterprise-wide representation of the information takes into consideration the requirement that the information must support multiple functions. One result is a database that allows easy access and sharing of information across the enterprise. Another result is a set of applications that have little or no overlap.

For users of IBM systems, an early appearance in 1972 of an industry application architecture (though it was not called as such) was the Communications Oriented Production Information and Control System (COPICS).[3] The intent of IBM was to publish concepts, structures, and possible approaches that could be used by manufacturing companies in their applications. No applications accompanied the COPICS manuals in the beginning. IBM pointedly stated that COPICS was not a specific solution and needed adaptation to fit individual companies.

In 1977 IBM introduced the Planning Aid for Retail Information System (PARIS),[4] and in 1980 introduced the Grocery Information Processing System (GRIPS).[5] These documents aimed at the same goals for the retail industry as COPICS pursued for the manufacturing industry.

An example of an industry application architecture on which many organizations from several countries have worked together since 1984 is the ESPRIT Project 688 of the Commission of the European Communities: The Computer-Integrated Manufacturing Open System Architecture (CIM OSA).[6] The project currently consists of 22 organizations from 10 countries. CIM OSA defines the architecture for a CIM enterprise and intends to



provide building blocks for the design and execution of a CIM system.

Recent years have seen a number of announcements of industry-specific application architectures by IBM. In October 1989 the Computer-Integrated Manufacturing Architecture, which incidentally is based on CIM OSA, was announced, and in 1990 the Corporate Management Information Model designed for the wholesale banking industry, the Insurance Application Architecture, and RAA were announced. In 1991 the announcement of Computer-Integrated Logistics followed. Announcements for other industries are expected.

The concept of modeling. To arrive at an application architecture for the retail business, we start with the development of a model of the retail business system. Since there is no unique retail business, our model is to cover various kinds of retail outlets (e.g., supermarket, hypermarket, department store, variety store, specialty store) and various kinds of merchandise being sold by these outlets (food and nonfood). Thus our model is to describe a generic retail enterprise.

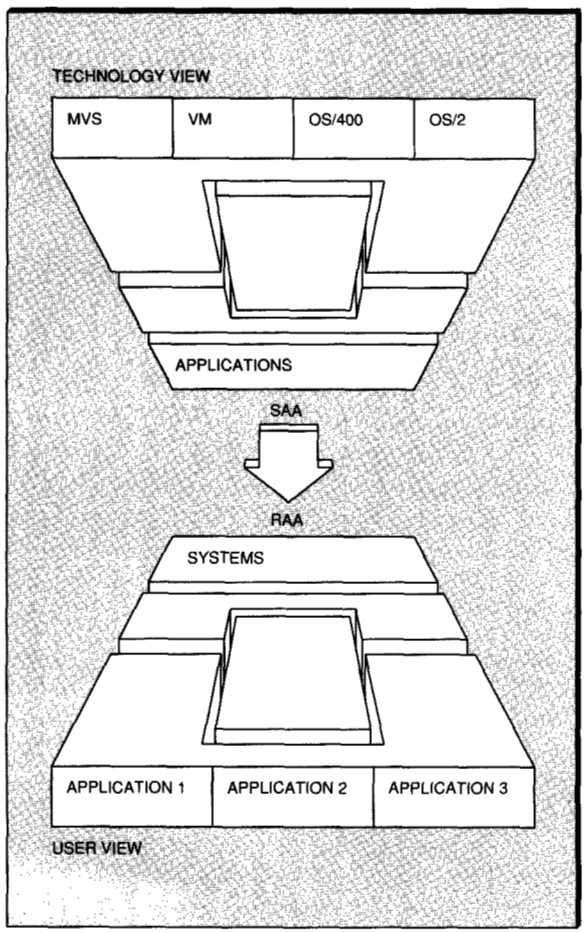
Enterprises are dynamic. Their processes, organization, and requirements are ever-changing. Enterprise modeling is the activity of defining how an enterprise operates. It is done to better understand the enterprise activities before any changes are undertaken. In most complex environments such as automotive or aircraft manufacturing, the need for modeling is commonly accepted as a means of simplification and concealment. For example, the development of a new car begins first with a model—often basic—of the car before a complete set of engineering drawings is produced. This first model allows the designers to get a feeling for the overall proportions of the car. This model becomes more precise over time until a first version of the car is built.

Similarly, it is possible to apply modeling techniques to a retail enterprise. At the higher levels it helps a retail executive identify the functions required to run the business. Models at this level cover the entire enterprise, not just the functions that are computerized. The enterprise is defined in business language, not I/S jargon. At this level, it is possible to consider different strategies for running the business and to assess the effect on the business processes.

As the model is refined and becomes more detailed, it is possible to consider the procedures and rules that govern processes, e.g., what needs to be done to have a new forklift in the warehouse. It is at this point that differences between a manual and computerized process appear.

In the RAA project we set out from concepts elaborated in a number of papers.[6,7–9] What these papers have in common is that they define a framework for relating a business system to its representations in a computer. The functioning business system is put at the top, and the functioning computer with its hardware and software is put at the bottom of the framework. For methodological reasons it is assumed that they can be separated, whereas in reality they are deeply intertwined. According to the application controller concept,[7,10] an installed data processing system is a mapping of a particular business system in the sense that certain activities in it that previously were accomplished manually or by special-purpose machines, or were not done at all because of the vastness of the problem, are now performed by a computer using programs, files, and a program schedule. As there is no way yet to translate

Figure 2 RAA—orientation



business system activities directly into programs, files, and triggers, this mapping has to be done in intermediate steps with the help of system analysis and design techniques. The business system level and the data processing level are "populated" by functioning systems, namely a business organization with its objectives, processes, policies, and entities, and a data processing system with its hardware and software systems, respectively. By specifying requirements and designing a data processing solution for business problems by any method such as flowcharts and decision tables, an additional level (an "interface" level) is introduced between the business system level and data processing system level and is called an application level. The application level contains a data model and a functional model. The functional model is built with four structural elements: activity, state, agent, and message. The application controller concept suggests using the functional model not only during the development of the data processing system, but also during its operations as the point of reference to business users and the data processing system, and to poll it continuously in order to identify those processes that are due for execution by a business user or the data processing system. The processes due for execution are put in a queue to be accessed by business users and processors. By incorporating driver and monitor functions for the applications, the application controller becomes an interface machine between the business system and the data processing system.

The Information Processing Architecture[8] starts with the so-called classes of business process, information, application, data, network, and support system (see Figure 3). The business process class and information class are derived from the business strategy and provide the basis for the application, data, network, and support system. The relationships between the different classes are shown in the figure.

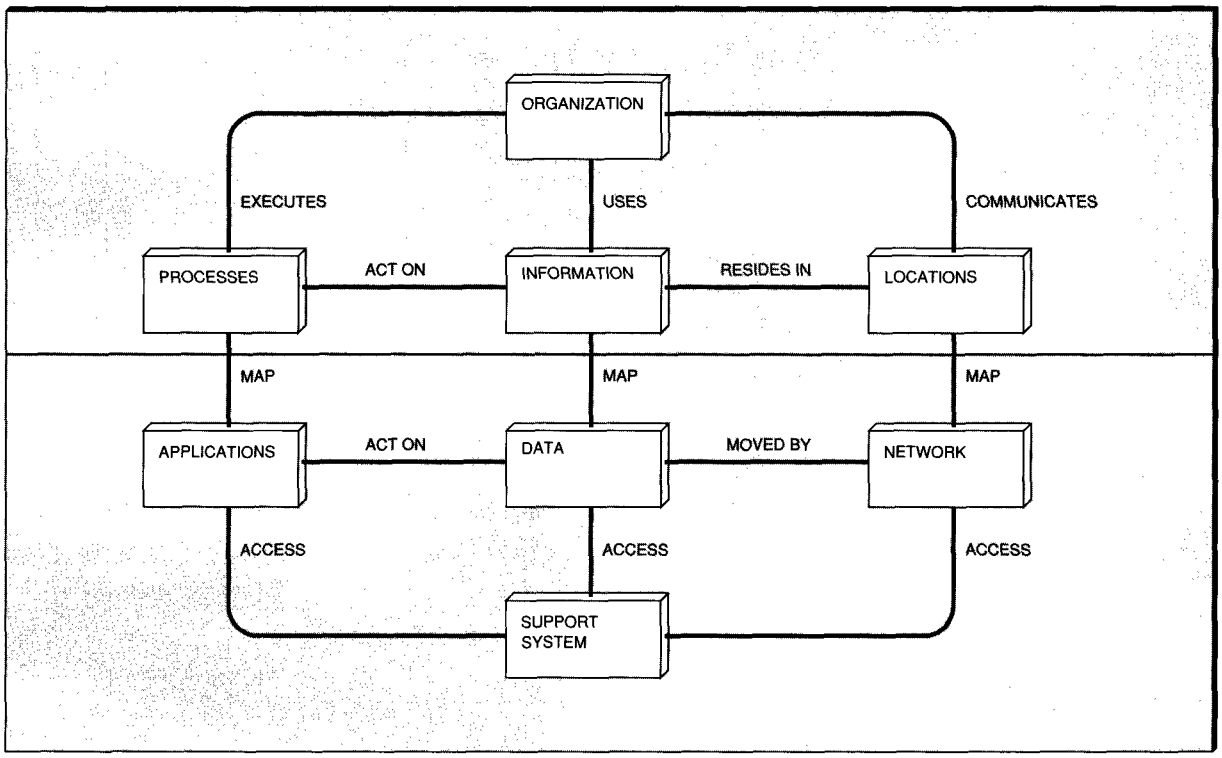
For his framework, Zachman[9,11] describes the creation of a product such as an aircraft or information system that is to be built by means of an architecture. He wants to represent the different perspectives of the players in the build process, such as the planner, the owner, the designer, the builder, and the subcontractor, and therefore defines the architecture levels of scope, business model, information system model, technology model, and detailed description, respectively.

The scope depicts the basic purpose of the final system, its performance, and the costs of building it.

The business model reflects the viewpoint of an owner, department manager, or business professional and contains the business processes and entities and how they interact. At the business model level, no reference is made to implementational aspects, for instance, whether a particular business process is carried out by a machine, e.g., a computer, or by a person.

The information system model takes into account the information aspects of the business model and deals with the data elements and functions that represent business entities and processes. The in-

Figure 3 I/P architecture model



Reprinted with permission from *Information Processing Architecture, Presentation Guide,* IBM Corporation, January 1985

formation system model specifies data entities and data elements, high-level application design and structure, data processes, data input and output to processes, data stores, etc. Again, no reference is made to implementational aspects. The information system model is intended for the application designer and data administrator.

Technology constraints are introduced in the technology model, e.g., hardware and software to run the applications and connect them. The technology model addresses the generation and execution of applications such as screens, reports, queries, program logic with constraints and derivations, programming languages, and physical databases. The technology model is aimed at the programmer.

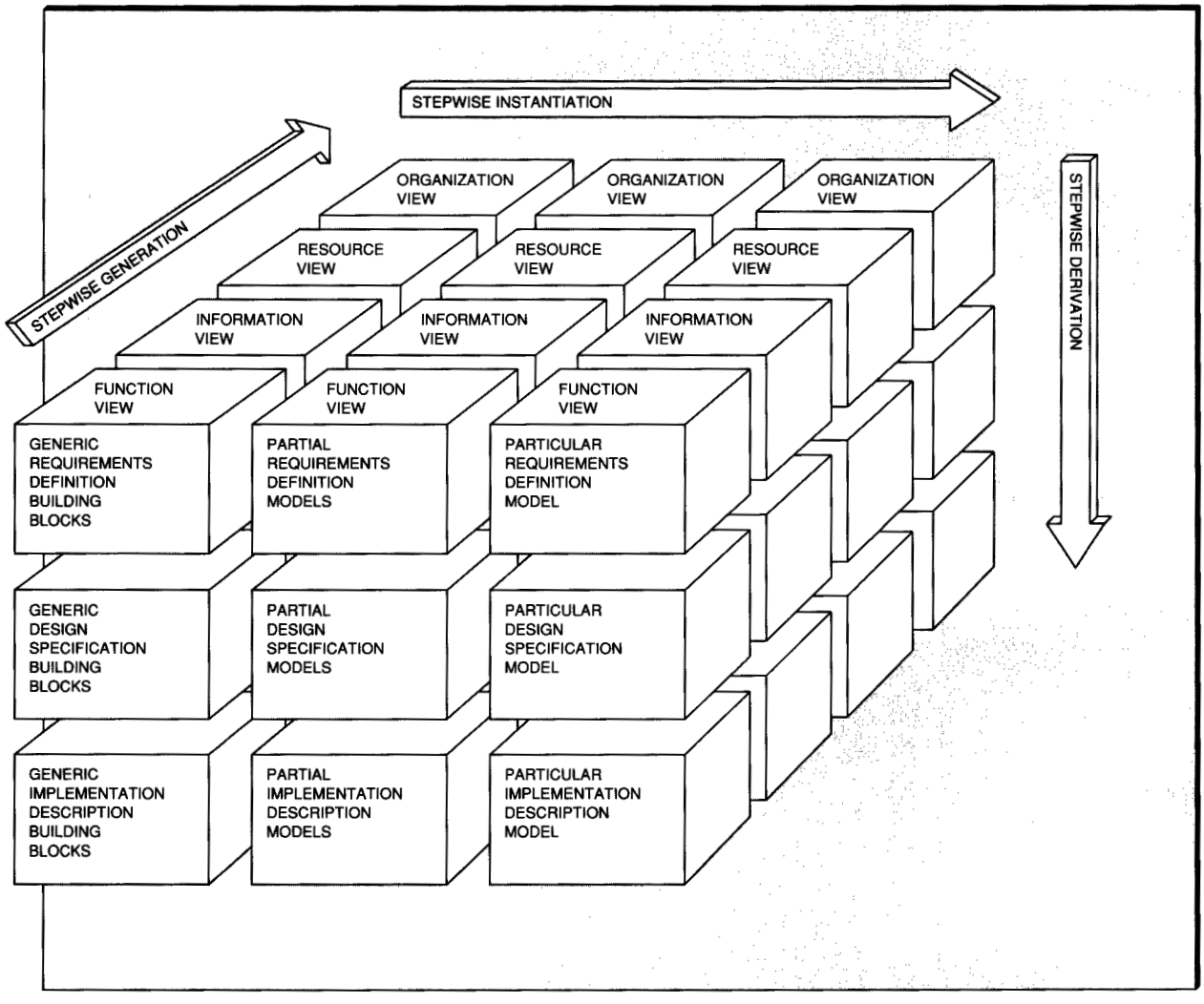
The detailed specifications make up the fifth level. They are given to programmers who code modules.

Zachman makes the point that each level has a different nature from the others: "They are not merely a set of representations each of which displays a level of detail greater than the previous one. . . . They are actually different representations—different in content, in meaning, in motivation, in use, etc."

The second dimension of the Zachman framework is made up of a set of so-called descriptions: data, process, network, people, time, and purpose. These descriptions are oriented to different aspects of the object described, such as material, function, location, responsibility, dynamics, and motivation.

The levels of business model, information system model, and technology model become the levels of requirements definition, design specification, and implementation description in CIM OSA[6] (see Figure 4). A second dimension addresses step-

Figure 4 CIM OSA framework



wise instantiation of the models: the generic level is a reference catalog of basic architecture constructs for components, constraints, rules, service functions, etc. The partial level is concerned with sets of partial instantiated models applicable to a sector of the manufacturing industry such as aerospace and electronics. The particular level is entirely concerned with one particular enterprise.

The third dimension of the CIM OSA framework defines "views." (Zachman uses the term "description.") Views are needed to fully model aspects of the enterprise: "The functional view describes the functional structure required to satisfy the objectives of the enterprise and the related control structure, i.e., the rules that define the control sequence, or flow of action. The information view describes the information required by each function. At the requirements definition level this is the business user's view of information. The resource view describes the resources and their relationship to functional and control structures. The organization view is the description of the responsibilities assigned to individuals

for functional and control structures, information, and resources."

When we compare these frameworks, the difference in the number of model levels between the functioning business system and computer system stands out first: the application controller has one level, the information processing architecture has two levels, the CIM OSA has three levels, and the Zachman framework has five levels. All frameworks cover the descriptions defined by Zachman except for the application controller, which does not address all aspects of organization; they may call them differently or consolidate some descriptions into a view. The number of levels and views becomes important when one considers the purpose of a framework. Zachman advocates the building of a model for each cell of his framework. (He calls an intersection of a level with a view a cell.) This directive leads to 30 models, a number far too high to be practical and useful, if one considers the purpose of modeling as a way to help in understanding complex issues. An opinion quite opposite would call for restricting the number of levels and models to a minimum and using a modeling framework as a point of reference to position various aspects of a model. For instance, one may say that a process decomposition relates to the functional view or that a data flow between two processes relates to the entity view. We shall repeatedly come back to the question of the "optimum" number of levels in an industry application architecture throughout the rest of this paper.

The application controller is the only framework in which the role of the model is to serve in application development and in the execution of business processes and applications. This purpose is achieved by defining the functional model as state networks. The CIM OSA framework is the only one that covers industry-wide and enterprise-specific aspects to a detailed level. The other frameworks are geared to developing an information system for a particular enterprise.

How can we succeed in representing an individual enterprise, a system that is so complex, diverse, and singular—not to mention how to represent all enterprises in retailing? This representation can, of course, only be done in approximation. The larger the set of enterprises becomes, the more unprecise the approximation will become. The challenge in modeling is to strike a balance between generalization and specifics: The models must be general enough to be valid for a wide range of enterprises, but specific enough so that the effort of designing a particular business system or application is limited. However, a certain effort for customization will always be required. This role is the one a consultant has to play in application architecture. Finally, we offer a word of caution. We may call aircraft design and enterprise modeling both modeling. We must, however, not lose sight of the fundamental differences that lie between them. An aircraft can be "frozen" in time and space, whereas an enterprise, like any social organization, cannot. It is recreated every day. The way in which processes are carried out and procedures are followed changes continuously, sometimes without the persons involved even noticing it. Because processes and procedures change continuously, any procedure manual is out-of-date on the day of its publication and many an application does not meet user requirements on the cutover day. The goals that we therefore defined for RAA had to be phrased with these fundamental limitations of enterprise modeling in mind.

RAA philosophy, components, and usage steps

In this section we first define the goals of RAA. These goals bear on the development methodology adopted in RAA and the components of RAA. We finally describe the basic steps for using RAA to analyze a business system and develop applications.

The RAA goals and methodological approach. Even 20 years after the first attempts were made to understand it, the concept of an industry application architecture is still new, and no approach or methodology has established itself. When we started the project of developing RAA in the second half of 1989, it was only reasonable that we adopt the attitude to learn by doing and include retailers and application vendors in the RAA project as a source of architecture specifications, both as coworkers and sounding boards. Before we decided on what we were going to develop and how we were going to do it, we first defined the goals of RAA:

1. RAA was to define a framework for integrating applications and data in a retail enterprise. The

applications either exist or are to be developed.
2. RAA should be a tool to be interpreted and used by analysts and consultants, not an all-embracing system to be followed meticulously and blindly.
3. As a tool RAA was to be employed in different methodologies to develop applications.
4. The framework had to be comprehensible to both business and I/S users.
5. RAA had to be industry-wide and cut across different retail outlets and national borders.
6. It was essential that retailers and application vendors would participate in the project.

We also wanted to come out quickly with first results that would then be modified and extended, as validation with retailers, application package vendors, and industry experts would suggest.

As these goals show, our initial intention was to build a framework for application development and integration. As the first models evolved, we recognized that RAA had validity beyond applications and databases. We could also use it to analyze a business system, re-engineer it, or define business or I/T strategies.

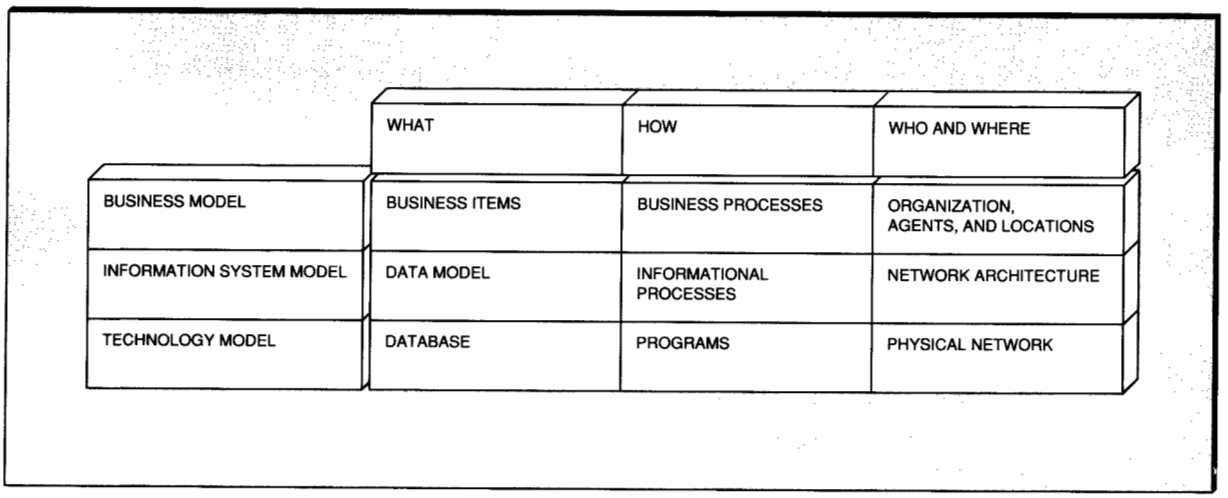
Once these goals were stated, the next step was to define the methodological approach we were going to take for actually developing RAA and which would meet the goals.

The RAA methodological framework. In the selection process of methodologies we were guided by the principle that any methodology has to be subject, even subordinate, to the problem it is intended to solve. Problem and methodology belong to two different worlds. The problem world is the world in which we live and try to survive. The things in this world—be they natural, such as human beings, or artificial, such as enterprises—are complex and whole. That we "model" human beings by means of concepts like body, mind, and soul and enterprises by means of concepts like entity, process, location, organizational unit, people, and time is a matter of convention and convenience. Concepts belong to the methodology world. They are an abstraction from the real world and as such are incomplete. A look at the concept of organization will show their incompleteness. No matter how elaborate an organization chart of an enterprise is, it does not capture the multitude of formal and informal dependencies that exist among people in an enterprise. A methodology aims at being consistent in itself; it thus resembles mathematics. The only reason for pursuing methodologies has to be for their convenience in solving problems. Convenience is determined by the closeness of the concepts to people's views of a problem and the effort required to solve it.

We started with the Zachman framework but were going to use it as a point of reference, not with the intention to develop models for all cells. The models we had in mind were to cover business entities, data entities, business processes, informational processes, and organization. We were also conscious of the fact that the Zachman framework is meant for developing specific applications, not an industry-wide application architecture. This had repercussions particularly for the technology model level and to a lesser degree for the information system model level. The technology model level poses a real problem to an industry-wide application architecture, since we cannot possibly represent all implementational aspects of applications in a whole industry, such as data and logic specifications and distribution of processors. The higher model levels are more abstract and therefore easier to handle. We had to find another way of coping with the technology model. We decided to leave the detailed specifications to application packages and "tie" them to the higher levels by a set of rules—the compliancy rules. Generally speaking, an application is compliant with RAA if it can be positioned in the RAA models. (The subject of compliancy will not be discussed in this paper.) If in the Zachman framework we consider the business model and the information system model together as a model of the enterprise, we can then view RAA as consisting of an industry-wide enterprise model and compliancy rules that define the way in which applications implement the enterprise model. The enterprise model and the compliancy rules together allow us to rightly call RAA an application architecture.

Views within the RAA methodological framework. Once we had decided to focus on the business model and information system model levels, the next question to be answered was which views of the enterprise to select. The frameworks introduced above refer to concepts like entity, data, message, resource, process, activity, action, function, agent, organization, location, and state. Other concepts that can be found in the literature and which we may call views as well

Figure 5 An architectural framework



Adapted from J. A. Zachman, *IBM Systems Journal,* Vol. 26, No. 3, 1987

are: objective, reason, goal, critical success factor, object, and event.

The basic views with which we started in RAA are the following (though these concepts are inspired by Zachman, they are proper RAA definitions). (See also Figure 5 and Figure 6).

Entity is oriented to things that an enterprise wishes to be aware of, such as persons, money, goods, or data. Special kinds of entities are the resources used or consumed by the enterprise in the fulfillment of its objectives. Examples are: product, fixed asset, personnel, money, customer, vendor, market, and funds invested in the enterprise by owners. Entity answers to the question "What is used in an enterprise?"

Process is a group of logically related decisions and activities required to manage the resources of the business. It has objectives and consumes resources. Its performance can be measured. Processes transform entities. This transformation happens in processes such as distribution of goods and calculating sales totals from individual sales transactions. Process answers to the question "How does an enterprise operate?" We use the terms process and function interchangeably, with a slight preference for process since some people understand function to be an organizational unit such as the sales department or warehouse organization. Also, we agree on the convention that a business process can be performed by a person or a machine and can be at any level of detail, even at the program module level.

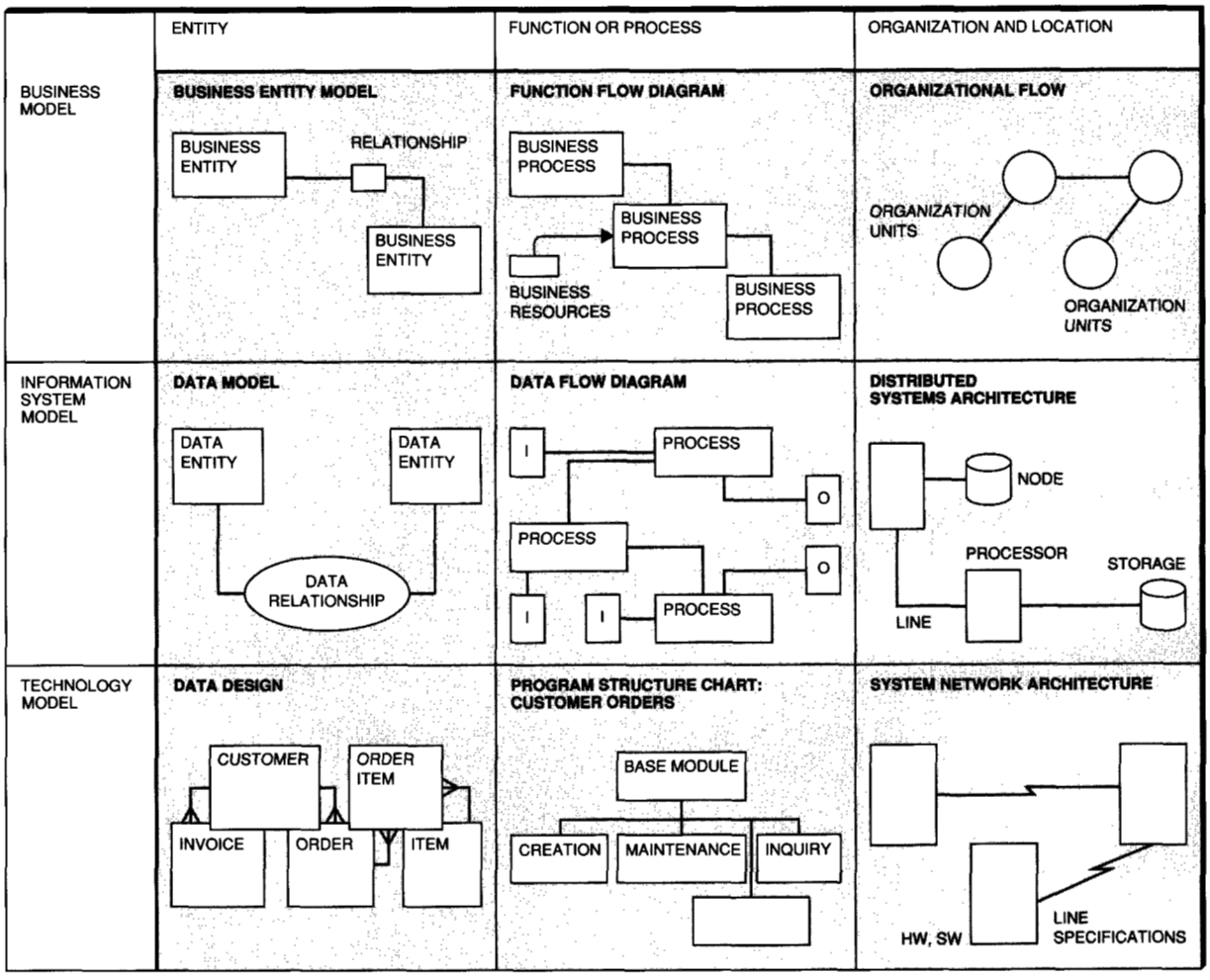
An organization defines responsibilities assigned to individuals to perform business processes and achieve business objectives. These individuals are called agents. Organization and agent answer to the question "Who carries out activities in an enterprise?"

Each agent operates in one or more locations. Location also specifies where processes are carried out or entities reside, e.g., headquarters, store, or warehouse. Location answers to the question "Where do things happen in an enterprise?"

Event defines when processes occur. It may be a point in time, or an external event such as a customer entering a store, or an internal event such as running out of stock for certain merchandise. In RAA events are represented through triggers. Event answers to the question "When do things happen in an enterprise?"

Each one of these views occurs on each of the model levels: business, information system, and technology. Figure 6 is based on the Zachman

Figure 6 An architectural framework with examples of diagrams

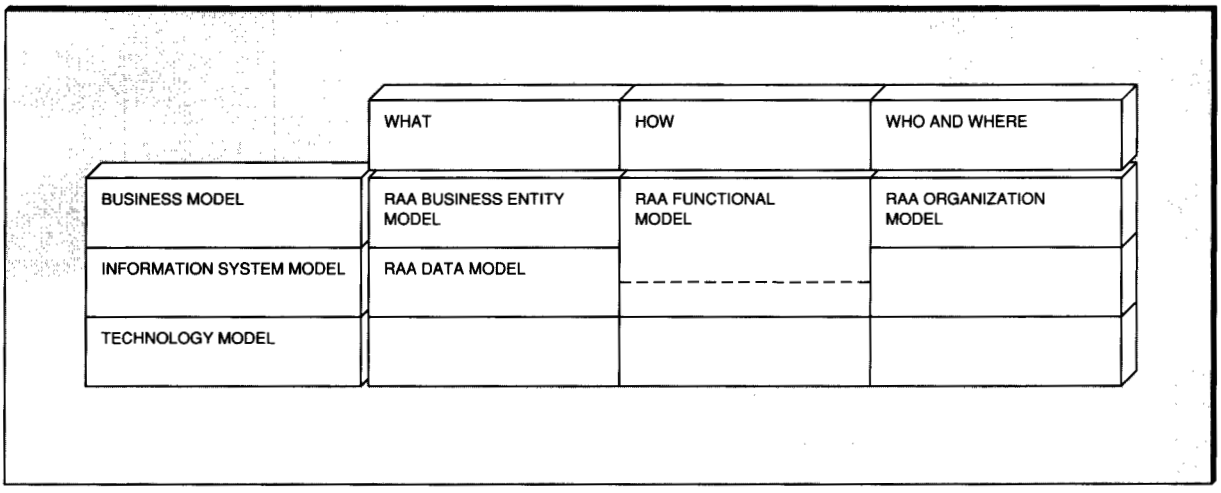


Adapted from J. A. Zachman, *IBM Systems Journal,* Vol. 26, No. 3, 1987

framework and depicts some of the aforementioned views. Note that organization and location are collapsed into one view. The reason for this consolidation is their conceptual similarity, once all of the processes performed by an organizational unit have been abstracted from it. An organizational unit is defined as a company or subdivision of a company, internal to the retailer, e.g., headquarters, buying department, goods receiving department. These units are just names that become meaningful only after the processes that occur there are identified. Once these processes are abstracted, what is then left is the name of a department, similar to the name of a location. A location indicates a geographical place, whereas an organizational unit denominates an "organizational place" in the hierarchy of an enterprise. Because both refer to places, we combined them into one view.

Tools versus techniques versus methodology. With the advent of tools for use on personal computers, the task of modeling has been simplified substantially. RAA was developed with the Application Development Workbench** (ADW**) of KnowledgeWare, Inc., an AD/Cycle* tool. The techniques employed in RAA, however, such as entity/relationship diagrams, decomposition dia-

Figure 7 The RAA enterprise model within the architectural framework



Adapted from J. A. Zachman, *IBM Systems Journal,* Vol. 26, No. 3, 1987

grams, and data flow diagrams were selected with the goal to store, customize, and enhance the RAA models by means of other tools available in the market.

A methodology combines a set of techniques to achieve a larger goal, such as business analysis or application development. We were eager to keep RAA as independent from any particular methodology as possible in order to enable a prospective user to select the methodology with which the user is most familiar or the one that would best help the user solve the problem at hand. Examples of such methodologies for application development are information engineering[12] and MERISE.[13] The freedom to keep RAA independent from any particular methodology is limited by the methodology inherent in the tool by which RAA is built. Information engineering, for instance, is quite weak in covering the dynamic aspects of systems like events, triggers, and states. Because ADW is based on information engineering, it is consequently not easy to represent triggers for processes within its diagrams; text or matrices are used for them instead.

The RAA reference architecture. After we reviewed the goals and methodological approach with retailers and application vendors, the development of the architecture commenced.

The goal of RAA has been to develop an application architecture not for a specific retail enterprise, but for a generalized retail business, cutting across lines of business such as hypermarket, department stores, and specialty stores, across types of merchandise sold, across organizational types, either centralized or decentralized, across locations such as warehouse, store, and headquarters, and across country borders. To cover such a wide scope, our architecture specifications had to be quite abstract. What we came up with was a reference architecture, similar in concept to the generic level of CIM OSA. Its intention is to describe the basic principles of retailing. The RAA reference architecture consists of

- Static functional model
- Business entity model
- Data model
- Organization definition

The static functional model addresses the view of process (how?) in the architecture framework, the business entity model and data model follow that of entity (what?), and the organization definition that of organization (who?). See Figure 7.

The RAA static functional model. The static functional model is abstract and universal, and defines the common activities that occur in a retail en-

terprise such as buying, storing, and selling merchandise. It describes what kind of processes exist, what kind of entities and information can flow between processes, and what event triggers the execution of a process. It does not specify where a process happens (location), who executes it (organization/agent), what technology is used, and in what logical or temporal sequence the process is executed. Because it disregards the logical and temporal aspects of processes, it is called the static functional model.

In the definition of business processes we were led by the concept of object-oriented design that groups functions by the main object with which they deal, and not by entity life considerations. The objects to start with were the resources in a retail enterprise such as product, client, and personnel.

A typical example for this methodological approach of RAA is the business process Determine Product Needs. The product needs determined in this process can be specific customer requests for a particular product such as a complete kitchen made to specifications. It also covers determining and processing orders by customers for items that are simply out of stock. It can be the sales forecast of items that are part of the standard product range. This forecast is then compared in this process with quantities on hand and on order, and quantities to be ordered are recommended. We recognize that these subprocesses of Determine Product Needs are triggered by different events, executed by different organizational units and agents, happen at different locations, follow different business rules and procedures, and generate different information flows, but are nevertheless part of the one process whose objective is to determine what the product needs are in terms of kind and quantity. Determine Product Needs exemplifies the object-oriented approach and illustrates what we mean by saying: the business processes of the RAA static functional model are abstract and universal. As we shall see below, we did, however, not follow the object orientation rigorously.

An important feature of the RAA methodology is that each process occurs only once in the static functional model. Picking of goods, for instance, happens at distribution warehouses, at buffer areas in a store, or from shelves in the selling area of a store. These processes are performed by personnel of the retailer and by customers. There is, nevertheless, only one process Pick in the static functional model.

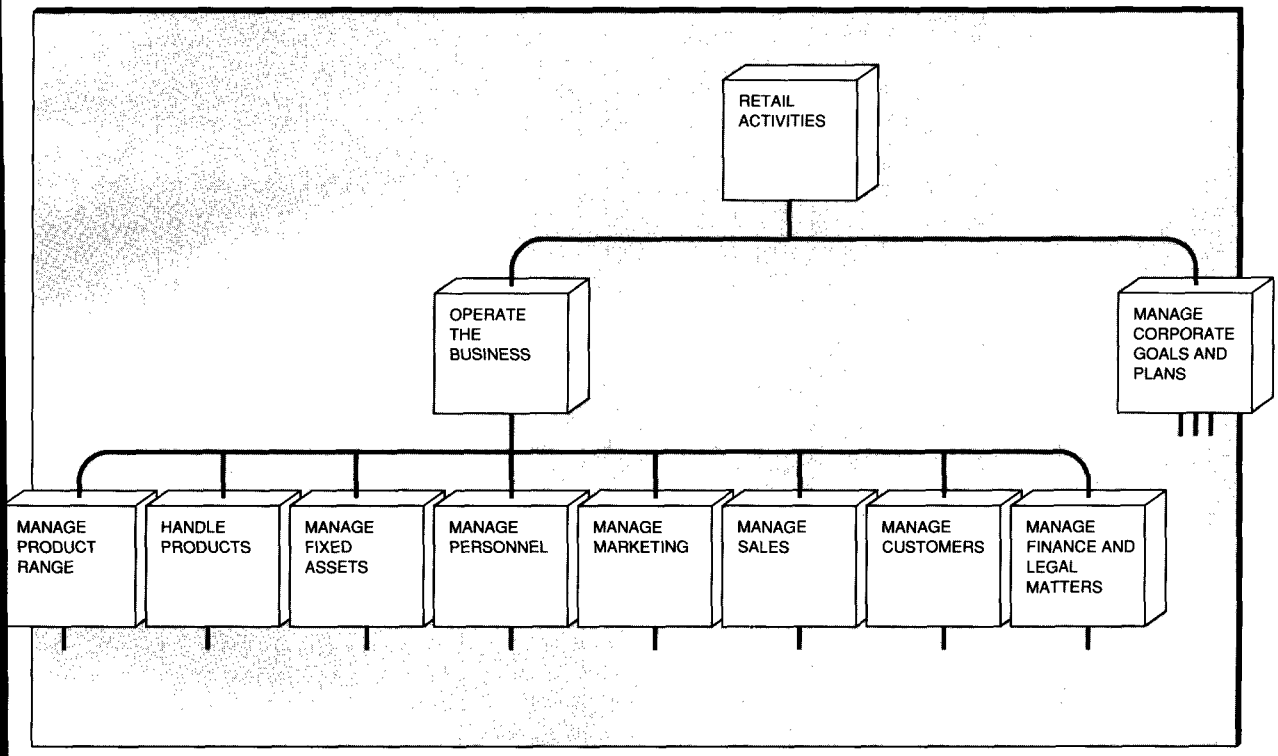
Another example of the universal and abstract property of the static functional model is the fact that we do not refer to stores and warehouses. The business processes are described in such a way that they are valid no matter where the sale to the customer is negotiated, or where the goods are stored.

To enrich the definition of processes we include the flows of entities and information between them in the static functional model. This is done by picking any two subprocesses of a given process and specifying all direct flows between them, i.e., flows that connect the two subprocesses and do not pass through other subprocesses. Flows may also exist with subprocesses in other processes inside or outside the retailer.

The retail activities are grouped into business processes at various levels of detail. At the high level, we have identified nine major business processes in a retail enterprise (Figure 8):

1. Manage Corporate Goals and Plans—This process concentrates on the general management, long- and medium-term planning, reporting analysis, organization, and strategy of the enterprise. It defines instructions that become input to other processes, for instance, by setting objectives. It receives from the other processes performance reports and resource requirements.
2. Manage Marketing—This process covers analysis of market needs and offers, advertising, and promotion.
3. Manage Product Range—This process covers the selection of the range of products to be carried; controlling inventory; buying and contracting; and ordering and pricing.
4. Handle Products—This process addresses all aspects of handling goods and preparing them for sale, from initial receipt to the time of sale.
5. Manage Sales—This process covers all aspects of selling goods and services to the customer, from assisting the customer in finding the goods sought, to taking payment.
6. Manage Customers—This process addresses all aspects of building up a long-term relationship with customers, for instance, creating an interest in the products that the company of-

Figure 8 Retail activities



fers by direct mail and managing credit available to customers.

7. Manage Personnel—This process deals with the human resource management of the enterprise, such as welfare, career, training, and personnel statistics.
8. Manage Finance and Legal Matters—This process covers all financial, accounting, tax, and legal matters, including the physical handling of cash.
9. Manage Fixed Assets—This process covers the procurement and maintenance of assets used or owned by the enterprise, such as buildings, equipment, and insurance.

The so-called retail cycle is split into the supply phase,[14] which consists of the selection of vendors and product range, of pricing, of vendor negotiations, of ordering products, and of receiving and distributing products, and the sales phase. The retail supply phase is covered in RAA by the business processes Manage Product Range and Handle Products, the sales phase by Manage Sales and Manage Customers. Manage Marketing relates to both phases.

What made us segment the activities that occur at a retail company into exactly these processes? The first major split was to separate all activities relating to long- and medium-term planning and general management from the day-to-day operational activities and group them into the process Manage Corporate Goals and Plans. The operational activities were then grouped by major resources. These resources were identified to be market, product, client, personnel, finance and legal matters, and fixed assets. Product is what the retailer sells and includes goods and services. Market and client, next to product, are the main focus areas of the retailer. The market consists of suppliers from whom a retailer buys and clients to whom the retailer sells. What then distinguishes the client part of the market from the resource client? The difference is that market is the ensemble of all clients, the "anonymous client," whereas client is the known person with whom a

retailer gets into contractual relationships. Why did we not introduce supplier as a resource and define a major process to manage the supplier? The reason was that we felt that the supplier part is well understood today, whereas the client part is still evolving and needed special attention. The processes regarding suppliers were subsumed under the processes regarding product and market. Traditionally market, supplier, and client have not been considered enterprise resources. But if we enlarge the concept of resources slightly we may very well include them.

As we detailed the activities related to clients we recognized that it would make good sense to separate the activities triggered by the retailer, e.g., communicate with client and manage client credit, from those triggered by the client, e.g., a client buying goods in a supermarket. We thus defined a business process entitled Manage Customer and another one entitled Manage Sales. Hence, the resource client became the resources customer and sales.

The activities surrounding the resource product are manifold. We have seen that those related to selling are grouped in the business process Manage Sales. The ones related to buying, storing, and distributing were split into activities that physically handle products, such as receiving them and storing them, and activities that deal with the logical aspect of products, such as calculating goods on order or in store, or ordering goods. The physical handling of goods is in Handle Products, the logical processing of goods is in Manage Product Range. In Manage Product Range there are also activities involving the resource supplier such as buying, contracting, and ordering. The reasons for splitting the activities associated with product into Handle Product and Manage Product Range were both conceptual and practical. Conceptually there is an essential difference as to whether one deals with the physical product, e.g., when one picks it, or whether one manages its nonphysical aspects like negotiating with suppliers or determining the selling price. Practically, the number of processes related to product is huge in retailing and did not fit very well under one major process.

In conclusion we see that we used the aspects of function and resource simultaneously in order to segment the retailing activities into business processes. The aspect of function dominated when we defined the following processes:

- Manage Corporate Goals and Plans
- Manage Product Range as a subprocess of Manage Product
- Handle Products as a subprocess of Manage Product
- Manage Sales as a subprocess of Manage Client
- Manage Customers as a subprocess of Manage Client

The aspect of resource was predominant for the following:

- Manage Marketing
- Manage Product as a preliminary process
- Manage Client as a preliminary process
- Manage Personnel
- Manage Finance and Legal Matters
- Manage Fixed Assets

The RAA static functional model uses decomposition diagrams and flow diagrams to represent processes. A decomposition diagram is a hierarchical tree of processes (Figure 8). A flow diagram depicts the same processes, in this case connected by arrows representing the flow of entities and information. An example of a flow diagram can be found in the Appendix.

The nine business processes presented above are decomposed into various levels of detail. The principles of decomposition are:

- A process is broken up into approximately equally large chunks of activities. Such a chunk will become a subprocess.
- Similar kinds of activities should be put together in a subprocess.
- Flows within a subprocess should be a maximum.
- Flows between subprocesses should be a minimum.

The level of detail to which a particular business process is decomposed depends on the complexity and importance of the business process. The processes dealing with products, for instance, are complex and important and are hence decomposed to a low level. The processes concerned with fixed assets and personnel, in contrast, are less detailed because we assume that being of a more general nature they are covered by general

enterprise models. The guideline for how detailed a business process should be in the static functional model is mainly defined by our goal to keep the static functional model universal. To obtain this goal, in general, specifics are not taken into account. These specifics stem from lines of business implemented in the retail enterprise, such as hypermarket, department stores, and specialty stores, or types of merchandise sold, or organizational types, or locations for warehouse, store, and headquarters.

As Figure 7 illustrates, we combine the functional models on the business model and information system model levels. Why? Zachman, for his part, is very strict in demanding functional models for each level in his framework. As a matter of fact, we started off with separate functional models on the levels of the business model and information system model. The main difference between them was that the latter contained data stores and information flows, but no entity flows. The processes, however, were identical because we did not want to preempt in any way which processes were to be computerized and which ones were not. After all, we were developing an industry application architecture. A data store indicates that information between processes does not flow directly but via a storage area. The reasons are twofold: either the information needs to be kept for legal or statistical purposes, or the information sink processes are not yet ready to receive, i.e., they are "out of synchronization" with the source process. We argued that the static functional model being abstract and universal should not be concerned with the synchronization of processes. (We shall see below that this is the task of the workflow examples.) A second look at the meaning of data stores for statistical and legal information revealed that they serve as storage areas as well. This time the synchronization gap is even wider. This result left us with just one difference between the two functional models: the flows of entities versus the flows of information. We saw no loss in precision if we represented them together in the static functional model. The static functional model therefore extends to the business model and information system model levels.

The static functional model does not address the subjects of:

- Detailed procedures on how to carry out processes
- Logic specifications, be they for derivation or decision
- Input and output specifications, be it screen or report layout

These items are absent because we want RAA to be an industry-wide framework, and we cannot hope to cover all detail procedures or logic specifications that exist in the industry. We say that they are an integral part of the application packages tied to the enterprise model by means of the compliancy rules.

Because we do not address the subject of detailed procedures and logic specifications, RAA is nonexhaustive as far as business rules are concerned. Examples of such business rules are: (1) we will not manufacture goods but only trade them; (2) we will have warehouses as part of the supply chain, rather than have manufacturers and wholesalers deliver goods to our stores directly; (3) we will pay our suppliers 45 days after receipt of the invoice. Nevertheless there are various ways in which higher-level business rules are represented in RAA today:

1. Entity/relationship models define them through their relationships and cardinalities. For instance, the relationships "buys," "sells," and "manufactures" between the entities of organizational unit and product signal that the retailer not only trades in but also manufactures products. Cardinalities define business rules as well, e.g., the m to n relationship between product and buyer specifies that a particular product may be managed by more than one buyer and that one buyer may manage more than one product.
2. Flow diagrams can describe business rules implicitly by specifying the logical sequence of operations. An example can be found in the Appendix.
3. Text can accompany the diagrams and give examples of business rules.

We may say that the static functional model is a shell. Like a shell it structures a space, the space of activities in a retail enterprise. But it allows the inner space to be filled with a great variety of detailed procedures about which the shell does not care as long as they fit into the boundaries of the shell.

The RAA data model and business entity model. The RAA data model and business entity model define the common entities that occur in a retail enterprise. An entity[15] is a class of things of which an enterprise wishes to be aware, to track, or manage individually, and about which the business therefore needs to keep information. An entity may be concrete such as item, employee, and supplier, or it may be abstract such as agreements. The relationships these entities can have are also defined. Examples are "item is supplied by supplier" and "employee orders item." The RAA data model and business entity model make up the RAA entity/relationship models (E/R models). The RAA E/R models use the well-known E/R techniques to represent entities and their relationships. They describe what kind of entities exist and the relationships that they have with other entities. They do not specify where an entity exists (location), who is the owner (organization/agent), what technology is used to handle it, or in what business processes the entity is used. The goal for the business entity model is to define entities on a high level with the business view in mind. (Thus there are no associative entities in the business entity model.) Examples are the negotiations between a buyer of an item and its vendor regarding prices and conditions. These conditions can be diverse and can affect, for instance, price ranges, depending on quantities bought by period, payment, and delivery and will not all be captured in electronic form. Hence, they do not appear in the RAA data model.

Entities in the data model relate to the informational aspect of objects in the business. The RAA entity/relationship models are much closer to the implementation level, i.e., the technology model, than is the static functional model, as is shown in Figure 7.

An entity is characterized in terms of its so-called attributes (or data elements) and the values these attributes can assume. Examples of attributes are name, number, price, weight, size, and color. Attributes are kept in the data model only. Attributes that are derived data such as average and totals are generally left out.

Our starting point for the development of the RAA entity/relationship models was the set of resources that helped us shape the static functional model, namely product, fixed asset, personnel, money, customer, vendor, market, and funds invested in the enterprise by owners. The model that resulted from them was found to be trivial. We then proceeded in a bottom-up way by defining the entities that make up these resources. For example, we "broke up" product into base item, item variant, item classification, item group, etc.

To get some order into the set of entities, and also to help in the creative process of finding entities usually not defined in a data model, we used four entity type classes in the RAA business entity model.[16] Each class represents a group of entities with common characteristics. The four entity-type classes are:

1. Partner—A natural or legal person, public institution, internal or external organization, or organization unit relevant to the company. A partner has an active nature and can carry out agreements. Examples are supplier, customer, and organizational units such as store or warehouse.
2. Object—An object can be material or nonmaterial. Examples include item, selling equipment, and storage space. Objects cannot close agreements.
3. Agreement—A mutual understanding between two or more partners governing the way in which business is conducted. It states the obligations of one or more partners (for example, buying, selling, and transportation agreements) and generates single occurrences of activities, so-called transactions. Regulations from governments or from a company itself are other examples of agreements.
4. Transaction—A transaction is relevant to the fulfillment of agreements and consists of executed activities. Partners are the main initiators of transactions. Examples include a purchase order issued by a buyer, merchandise arriving at a store, an invoice arriving at the accounts department, and a sales transaction.

These classes bring some order into the vast variety of entities by pointing to certain of their characteristics and their interrelationships. Partner corresponds to the RAA concepts of agent and organization unit. Partners use still entities, the objects, while they get on with their "routine" work, the transactions. Agreements underlie transactions and are a kind of wider contract, as opposed to the more narrow-focused transactions.

The entity-type classes help considerably in clearing up important aspects in a retail enterprise. The terms headquarters, store, and warehouse, for instance, are employed indiscriminately to denominate a partner and an object. As organization units, i.e., partners, they can do business such as buying and selling, and own products. As locations, i.e., objects, they can "only" store products.

By means of the business entity model, an RAA consultant will elicit information from a business user about the kind of work the user is involved in, for instance, what kind of agreements the user can undertake with internal or external partners.

In the static functional model section we saw that the RAA business processes are abstract and universal. We find similar features of abstraction in the RAA entity/relationship models, too.[17] Examples of such universal concepts are item, partner, supplier/receiver, order agreement, order to supplier, deliveries and receiving, invoicing, and storing location.

Item comprises a product or service for sale by the retailer or for internal consumption. A partner can be a legal person, public institution, internal or external organization unit, e.g., vendor, customer, buying department, or store. The entity partner groups together attributes that are common to all kinds of partners. Attributes describing specifics of one kind of partner belong to the entity of this kind of partner. This structure of attributes is similar to the hierarchy of inheritance in object-oriented design.

In retailing, concepts like vendor, store, warehouse, and customer are in most cases viewed as very distinct with no similarities between them. In RAA we emphasize their similarities. First, they are all partners. Second, vendor, warehouse, and store all supply items. The vendor supplies warehouses and stores, the warehouse a store, the store a customer. We therefore introduce the entity "supplier" to capture the characteristics of the supplying partners. When items are supplied, the receiving partner then becomes the receiver of items. This concept of supplier/receiver can be enlarged to include the reverse direction in this chain; for instance, when a customer returns a defective item, we can consider that customer to be a supplier to the store.

In addition to item and supplier/receiver, we define in RAA a number of universal concepts that relate to the transactions that happen between suppliers and receivers. Order agreement with its entities defines the often very complex conditions of sale by a vendor to the retailer. Initially devised for the transaction between an external vendor and the centralized buying department of the retailer, it is now also applied to any order transaction in the supply phase, from vendor to warehouse or store, from store to customer. The same holds for the specific orders that follow after an order agreement is closed. The pertinent entities can be used for orders to vendors or for orders from customers. Deliveries and receiving denotes all entities related to delivery of items, no matter where it happens. Invoicing is defined with the same entities wherever invoices arise in the supply phase. We also use the same kind of entities to denote locations where goods can be stored, be it shelves in warehouses or stores.

We must keep in mind that abstraction can be maintained only to a certain level of detail. Further down in the hierarchy when we come to implementation, the differences in the phenomena being described will become apparent.

A data model (and business entity model for that matter) always depends on the scope of the business processes it tries to include. Hence, a data model is not completely defined if the processes to which it relates are not given. The data model for the process Manage Product Range, for instance, differs from the one for Handle Products. This is not surprising in view of the fact that some entities only occur in one process, e.g., equipment is an entity employed in the process of Handle Products only.

The link between the RAA business entity model and static functional model is established through the mapping of entities and relationships to business processes; for each business process, all entities and relationships that occur in it are listed. We noticed that business users prefer to think in terms of processes.

The link between the RAA data model and static functional model, in contrast, is achieved by means of a mapping of entities and relationships to flows between business processes; for each flow all entities and relationships that occur in it are listed. If we now take all flows that occur in

a given business process and collect the entities associated with each flow, we obviously arrive at the set of entities of the given business process. That is, the information we receive from linking the data model with the static functional model is richer than in the case of the business entity model and static functional model.

The quest for industry-wide functional and organization models. Most of the projects of industry-wide application architecture we have studied focus on the data model as the main integrator of applications. The rationale given is that a data model is less prone to changes than a functional model, even that it is not viable to develop an industry-wide functional model in view of the huge variety of business processes. Hence, they first develop a data model. In a later step, a functional model is derived by adding processes, triggers, rules, etc., to entities of the data model. It is our belief that an industry-wide functional model can be built as well as an industry-wide data model by abstracting from specific cases in an appropriate way. The RAA static functional model tries to achieve this by, for example, abstracting from specific organization units and locations such as a store and warehouses.

In a similar way, one could envisage the introduction of a reference organization model with industry-wide applicability. Such an organization model would need to abstract from specific organization structures and job descriptions and restrict itself to the basic capabilities and responsibilities of individuals to perform business processes and to achieve business objectives. Among the basic capabilities we find physical capabilities, e.g., to move goods and process raw material; mental capabilities, e.g., to calculate, negotiate, and make decisions; and communication capabilities, e.g., to motivate people, to sell to customers, and to lead people. A capacity is associated with each one of these capabilities, e.g., a person can move goods up to a certain weight and during a certain number of hours, or lead up to a certain number of people.

Many of these capabilities can also be performed by machines, including computers. We call an individual or machine thus characterized, an agent. An agent combines various basic capabilities when the agent performs activities and business processes in an enterprise. Examples of human agents are a cashier, who is in charge of checking payments received or operating a point-of-sale terminal to conclude sales transactions with customers, or a sales department manager, who is responsible for a particular department. The manager's duties relate to sales, personnel management, department layout, and handling of goods in the selling and storage spaces.

In RAA we have taken up the idea of a reference organization model by giving job descriptions for many human agents but have not yet explored the subject of basic capabilities of agents.

The basic steps in the use of RAA. As we have seen, the RAA reference architecture consists of (1) static functional model, (2) business entity model, (3) data model, and (4) organization definition.

The reference architecture is abstract and universal in order to cover a variety of specific retail enterprises. Interestingly enough the static functional model is further away from the implementational level than the entity/relationship model and requires more effort by the developers to obtain a particular business system from it. With little effort, a database can be defined from the data model to become the center of the retailer's operational application or decision support systems.

Though the static functional model is abstract, it can be employed without further customization in a particular retail enterprise in numerous ways. It allows us to identify functions that are common across an organization or across applications. It can serve as a framework to analyze strengths and weaknesses of a business. For instance, it helps answer such questions as: Why does it take us so long to get the goods from the warehouse to the shelves of the store? Why is the error rate so high in delivering the goods our customers ordered? Why is our average inventory so much higher than that of our competitors? Once the weaknesses of the business are identified, the static functional model allows the retailer to redesign business processes and set priorities for which business areas to improve, to invest in, or to dispose of. Different options of running a business process can be simulated before implementation. Hence, the static functional model is also a framework for defining a business strategy of an enterprise. An application that runs today, or one that the retailer wants to buy or develop, can be

positioned against the processes and data flows of the static functional model. Once the business strategy has been defined and user satisfaction with the current I/T of the enterprise has been measured, a strategy can be drawn up by the I/S department for where to invest, be it hardware or software.

A principle of RAA is to keep the RAA static functional model unaltered and use it as a point of reference. It is the task of RAA consultants to develop the so-called workflow model for a specific retail enterprise. A workflow model is the very opposite of the static functional model. It is concrete and represents specific cases of how retailing is done. A workflow depicts activities as they happen in pursuit of an actual business goal and specifies which events trigger a process, what completion criteria there are, where a process happens (location), who executes it (organization/agent), what technology is used, in what logical or chronological sequence it is executed, and what resources (e.g., product, time, personnel, funds) are consumed in its execution. By including both business and I/S elements the workflow model extends to the business model and information system model levels, similar to the static functional model. As in the Zachman framework, there are two functional models for the business model and information system model levels. The cut of the models across these two levels, however, is completely different.

The workflow model uses the processes and flows in the static functional model to represent how specific activities are performed. The representation is done through diagrams, correlation matrices, and text. In the Appendix we show the workflow for sales order and delivery. It defines the rules and sequence of processes that govern the sale of a product to a customer who enters a showroom and selects a made-to-order product such as a customized kitchen. Sales order and delivery is a workflow example that is provided by RAA. The RAA workflow examples cannot, of course, define cases for all variations of retailing. We rather take typical examples such as how a customer buys goods in a supermarket or how a buyer decides about the range of products to be carried.

A consultant is free to choose the level of concreteness to which a workflow is defined. A workflow may consist only of a flow diagram and text defining the activities and their sequence, or it may be comprehensive and specify in addition, start and completion criteria, locations, organizations, technologies, and resources.

Similarly, RAA contains a number of organization examples that relate organization units to business processes and locations. The organization examples and workflow examples are introduced to facilitate the task of customization (see Figure 9).

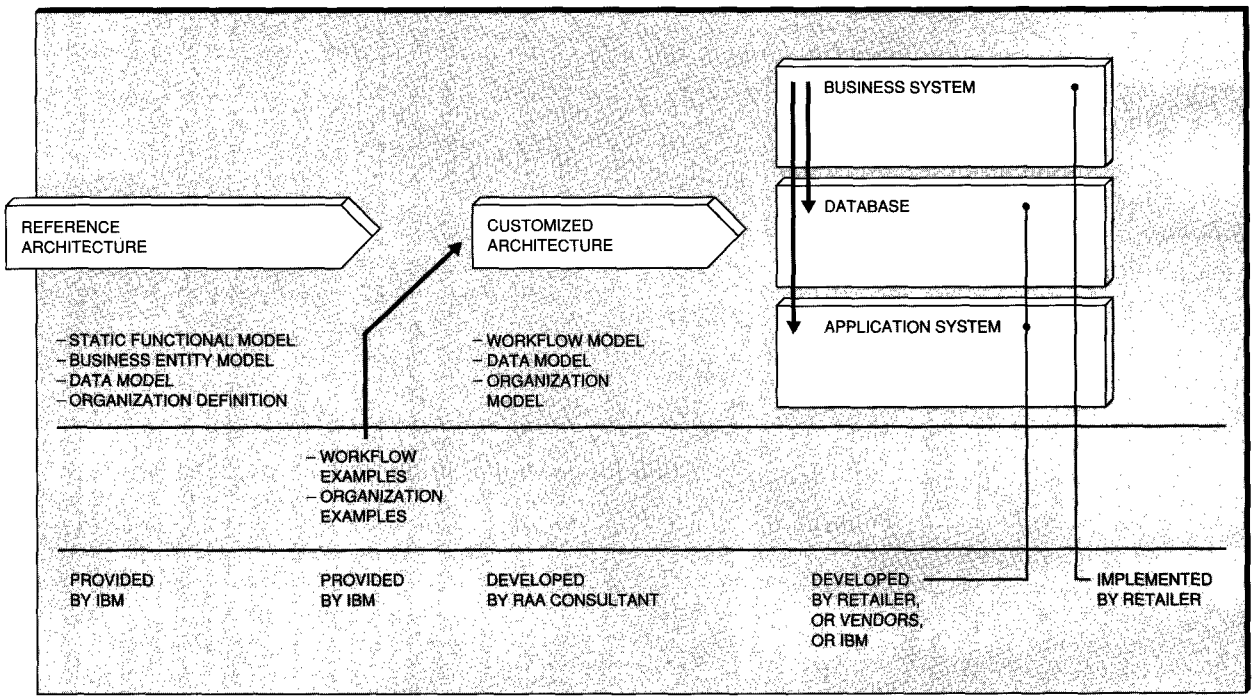
RAA uses the workflow examples and organization examples to bridge the gap between a generalized model and specific models of retail enterprises. In this respect we take quite a different route than CIM OSA, which introduces a partial level between the generic level and the particular level. The partial level contains sets of partial instantiated models applicable to a sector of the manufacturing industry such as aerospace or electronics.

The analysis of a particular enterprise is best started with the workflow examples and organization examples. Starting the analysis in this way will ease the understanding of the static functional model that is then employed to explore how business is actually done at the retailer across the scope of activities and organization of the retailer. A consultant uses the static functional model and the general business goals of the retailer in order to find out about the business rules of the retailer. These business rules, the static functional model itself, and the organization examples and workflow examples given in RAA allow the consultant to generate the particular business processes and organization model of the enterprise.

The outcome of customization is a model of the whole or part of the enterprise, its business processes, its entities, its flows of entities or information, and its organizational structure. We call it the retailer's enterprise model.

The customized architecture, i.e., the workflow model, data model, and organization model developed by a consultant, becomes the basis for a new business system with re-engineered business processes and organizational structures. It can also be the basis of a retail database, decision support system, or an application (see Figure 9). In order to go from the customized architecture to an application design, a number of items need to be specified, e.g.,

Figure 9 From reference architecture to customized architecture and implemented systems



- Distribution of application functions, for instance, cooperative processing
- Application flow of control
- Logic specifications, be they for derivation or decision
- Input and output specifications, be it screen or report
- Database specifications
- Data distribution
- Traffic over networks

What is RAA good for?

In the previous sections we defined the philosophy, components, and general steps in the use of RAA. We adopted the view of a system builder who wants to explain the system he or she has built to an engineering community. The main question the outside world will have in mind is "What can one do with it?" The three major areas for which RAA can be used are in the area of business systems, applications, and databases.

Business system. The use of the architecture specifications for business analysis and re-engineering is based on the fact that it is easier for an individual to describe what is right or wrong with the business by using predefined processes and entities as building blocks. It helps the individual's imagination. As such these predefined processes and entities can be compared with Lego** blocks in a child's world; for example, these blocks stimulate the child's imagination as the child uses them to build a toy car. Similarly a group of people will find it easier to communicate when faced with the task of defining the business in which they are engaged.

Business analysis. During business analysis a consultant takes the current business system and tries to understand what it achieves, not just how it works. The consultant seeks underlying assumptions and determines value-add activities. The performance of current business processes is measured in terms of time, cost, capital, value, and quality.

As stated in the last section, business analysis starts best with the RAA workflow examples and organization examples to build the workflow model and organization model for the enterprise under scrutiny. The static functional model and

business entity model are used in this analysis as a kit of building blocks to support questions for business users, like:

- Who are the owner, participants, and beneficiaries of the business process Manage Customers? What is the organizational structure to support it? What control mechanisms are used?
- What are the implicit and explicit business rules of Manage Customers?
- What are its objectives?
- What is its measurement system?
- What technology is used?

The result of the analysis is a list of strengths and weaknesses of the enterprise. Weaknesses may have their origin in outdated or faulty technology. They may be due to organizational deficiencies such as imprecise definitions of objectives, fuzziness of organization alignment, lack of measurement system, ineffectiveness of control mechanism, or lack of communication between processes. Once these weaknesses are identified, they can be tackled in a business re-engineering project.

Basis on which to re-engineer business processes. A company is often like a buried city in which layers and layers of procedures have piled up over the years. Nobody really understands them fully. Sometimes they contradict each other. To find one's way in the organization can take years. The mode of operation is not "business as usual" but "firefighting." The challenge is to reorganize and simplify business and take full advantage of technological progress. Application architecture is an important enabler for this process of reorganization and simplification. Business re-engineering takes over where business analysis left off after the strengths and weaknesses of the company have been identified.

Re-engineering is the fundamental redesign of the business system to achieve dramatic performance improvements. Although theoretically it can be done without re-engineering the I/T of the enterprise, in practice it often affects information systems and applications of the business. Business re-engineering changes the way in which business is done, through simplifying work and disposing of activities and organizational layers. Taylorism (from the methods of Frederick Taylor) broke up larger activities into smaller ones and assigned each one to a particular organizational function. Re-engineering the business reverses this method and views business processes as a whole. Business re-engineering should therefore not be confused with automation, headcount reduction, or a quality program. Its consequences may at times be painful to an organization. From conception to implementation is rarely less than two years.

The static functional model and business entity model serve to experiment in thought, investigate alternatives, and communicate preliminary results to business users and executive management. The workflow models and organization models developed during the business analysis will finally be revised to become the model for the new business system. After executive approval this new business system can then be implemented.

Examples of successful re-engineering projects are the introduction of just-in-time manufacturing in Japan and processing without invoices at Ford Motor Company, USA.[18] The reorganizations at IBM since 1987 are also examples.

An industry-wide enterprise model is an enabler that offers an organization the opportunity to explore various new ways of doing business. In other words, the benefits do not flow from the mere use of enterprise models, but arise from the human, organization, and system innovations that are sought in its framework.

Applications. Today, applications are typically isolated or are overlapping within an enterprise. It is a result of the way in which applications have been acquired over the past 30 years. Even now, projects are assigned to developers with high priority and little is done to ensure that the final application code fits well into existing applications. The development perspective is usually very narrow in order to meet critical schedules. As a result, development costs remain high from project to project, and the maintenance costs often run higher than the original development costs.

Application development. The process of developing applications traditionally begins with requirements definition. Then follow the phases of analysis and design, produce, build and test, and, finally, production and maintenance. The IBM platform for application development is AD/Cycle.[19,20] The RAA models are geared to requirements definition and to the analysis and design phases. An important part of the AD/Cycle requirements definition is enterprise modeling. The

RAA reference architecture serves as a blueprint from which the retailer's enterprise model can be customized.

In the application analysis and design phase the customized architecture and business requirements defined in the previous phase are used to help develop application design information, e.g., functional contents of each application module, specifications of application flow of control, and database structure and content.

Application positioning. Existing or required applications can be positioned within the RAA static functional model and data model. Positioning is done by marking all business processes, data flows, entities and their relationships, and attributes treated by a given application (see Figure 10). The goals for doing so may be diverse:

- A retailer wishes to identify these applications—either installed at the retailer's sites or available on the market—which support a given business process. The intention may be to improve the performance of this business process through advanced applications or decision support systems.
- An application vendor wants to communicate to retail customers which business processes are supported by his or her applications. Or the vendor selects these business processes with low application coverage in order to enter the market with an application in their support.

Application positioning against the RAA models is also important for building applications from existing code.

Building an application from existing code. A wealth of retail applications are held by retailers and application vendors. These days, an enterprise will very rarely start from scratch when it develops applications. It will either take its existing applications as a starting point, or prototype, or select one from an application vendor that will be customized to its requirements. Either way, the RAA architecture specifications will be invaluable as a guiding post. A dialog can develop with business users and the application vendor to match application functions against requirements. In this process existing applications can be viewed as a library of reusable components. Those components meeting user requirements wholly or partially form the basis of the new application. As they are positioned within the architecture specifications, it becomes clear what is still missing and must be newly coded.

This approach of using existing applications as a starting point limits the risks that always exist when applications are developed from scratch; user requirements are difficult to extract anyway. When they are extracted from a starting point of zero, they become even more fluid. Requirements change as time goes by for a number of reasons. Among them, the business changes, and it changes more, the longer application development takes. Users understand their requirements better as they become familiar with the features of the application under development. Also, user requirements change as the application under development changes the environment from which the need for the application arose.

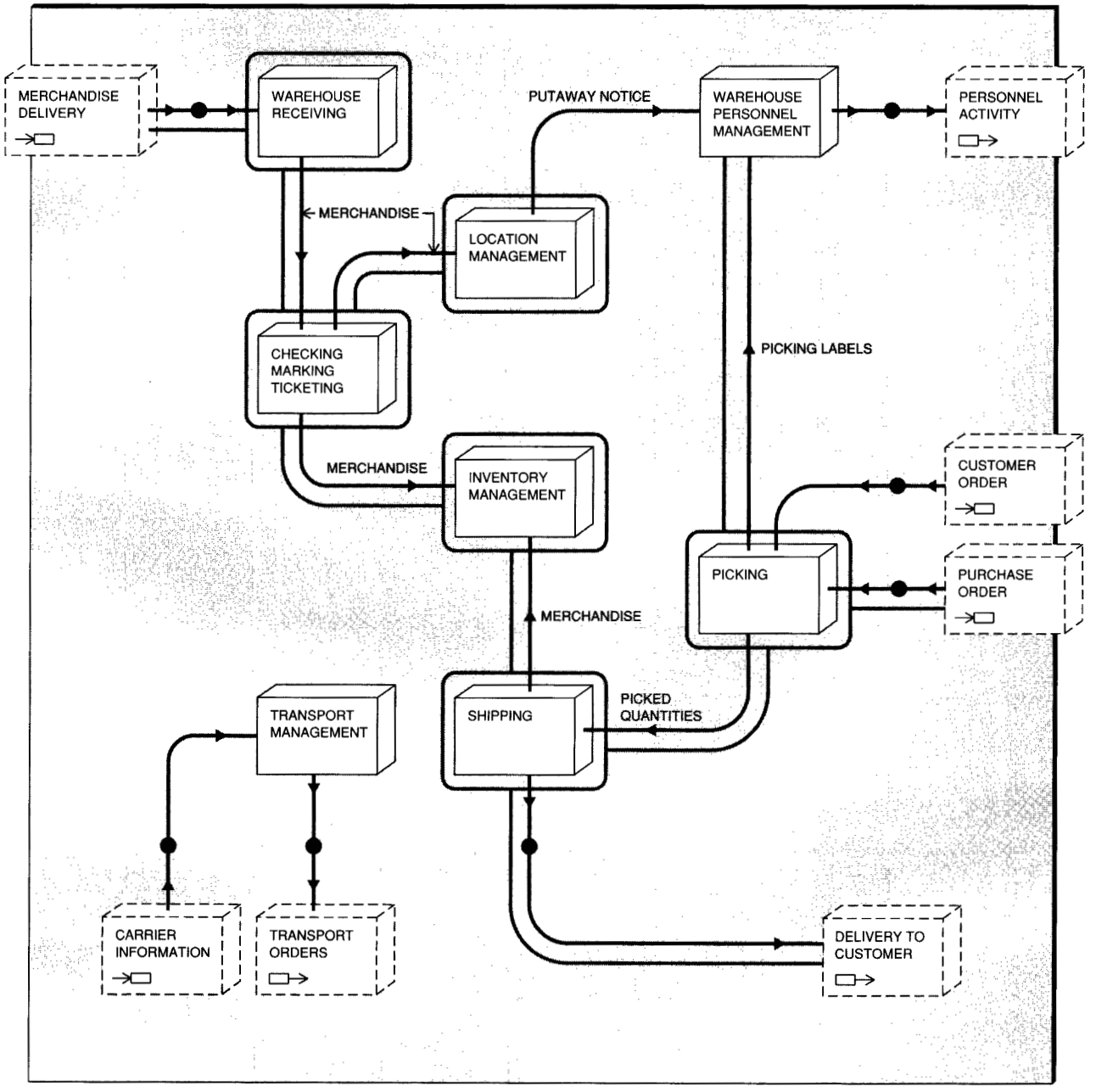
In principle, user requirements can never be stated fully before application design and coding start. To overcome this fact, it is less risky to begin with an existing application as a "library" of components that are being customized.[21] The vehicle to communicate with the business users during the process of customization is, among others, the RAA architecture specifications.

Database. The RAA data model is the basis for developing a relational database that can be "tuned" to a retailer's specific requirements, yet remain compatible with the RAA data model. This database can be in support of operational business processes such as Receive and Check and Prepare for Sale in Handle Products, or Process Orders in Manage Product Range. Or it can be built as a separate decision support database to be queried by business users regularly or *ad hoc*. Examples of business processes in which such a decision support database may be essential for business success are Select Products and Price Products in Manage Product Range. Such a database can well be linked to an expert system.

A data analyst can make use of the RAA data model for reviewing an existing data organization for possible inefficiencies.

Once a database is installed that corresponds to the RAA data model, an I/S organization can develop and install RAA-compliant applications more rapidly.

Figure 10 Positioning an application within the static functional model



RAA scope

An important feature of RAA is its abstraction from specific cases. We find it in the static functional model where, for instance, no reference is made to stores and warehouses, and in the E/R models where supplier can mean a vendor to the retailer, a warehouse, or a store. Therefore, RAA can be applied to nearly any form of trading in products, be it a store chain, or a single store

retailer, a retailer with or without warehouses, or one with a centralized or decentralized organization. More specifically, the RAA models cover the following for retailers selling through stores, with or without warehouses:

- All types of goods sold (food, nonfood, mixed)
- All types of organization (centralized, decentralized, mixed)
- All sizes of businesses
- All types of outlets (department stores, variety stores, specialty stores, supermarkets, hypermarkets, catalog showrooms, convenience stores, and so on)

We started the development of RAA with this scope in mind. After a first version of the RAA models had been generated, we were surprised to find that the models were also applicable, with minor additions, to wholesale enterprises, to wholesalers selling from warehouses, or to mail-order and home-shopping activities. In hindsight, this outcome seems to be only logical since our goal had been to abstract the essential features of retailing. After all, these features are not so different from those encountered in wholesaling, home-shopping, or mail-order businesses.

Not covered by the models in sales to the general public are hospitality (food and beverage, hotel industry), amusement parks, and purely service organizations (banking, insurance, dry cleaning, car and boat rental, travel agencies, and so on).

First experiences and next steps

When this paper was finalized in the fall of 1992, RAA had been used in a number of important projects at retail companies in Europe and Canada to analyze business systems, re-engineer selected business processes, develop I/T and application strategies in support of business goals and processes, perform data modeling, and teach I/S communities what the retail business is all about. The success so far has been encouraging. The goals we had set for ourselves before we started to develop RAA have been met, in particular, (1) RAA as a framework bridges the gap between the business user and I/S community, and (2) it is a useful tool when applied and interpreted by consultants with regard to the specific goals and methodologies of a project. The methodologies to carry out these projects differed—some used information engineering or its derivatives and some used locally developed ones. In every single case the RAA models fit in nicely. We think the fact that we built RAA as independent from any particular methodology as was feasible helped substantially in the acceptance of RAA. The consultants could continue to employ their favorite methodologies.

Though RAA is independent from any particular methodology, a common pattern seems to be followed, as suggested earlier. Interviews with the business and I/S communities are prepared with the RAA organization and static functional models. In the actual interviews the terminology and perspective of the interviewees are used. This use may even extend to a first cut of the workflow examples for the retail enterprise. It is in the analysis and workflow refinement phases that the RAA terminology and static functional model come into play fully.

Some detailed observations worth noting follow:

We were right in using our own terminology such as Manage Product Range, Handle Products, and Manage Sales, instead of the common terminology in the retail trade such as merchandise management, supply chain, and store management. Since nearly every retail enterprise or every person in the same enterprise understands something different by these "common" terms, it is easier to work with the "neutral" terminology of RAA. A side effect is that it often gives people a fresh view of the world in which they operate.

How many model levels should one keep in an industry-wide application architecture? There seems to be a trend to answer this question with one level. During the development of RAA there was a time when we had separate static functional models for the business model and information system model levels. In view of the fact that we were dealing with an industry application architecture where computerization of processes should not be preempted and data stores really had no place in the static functional model, we felt it was not worth keeping two different models, and we collapsed them into one. So far, a separate static functional model on the I/S level has not been missed in our projects. We were, however, somewhat surprised to realize that maybe we can do with one E/R model as well. The *raison d'être* of the business entity model was to communicate more easily with business users about entities with which they deal. This user group may find

the data model too technical. Consequently, the business entity model does not contain associative entities, combines certain entities into a super entity, e.g., storage place, warehouse area, overflow area, or reserved storage places into location. We tend to get two kinds of reactions from business users with the business entity model. For certain users it is still too technical since it uses the E/R diagramming technique. For others who tackle the hurdle of the E/R technique, it is not detailed enough, and they immediately proceed to the data model. If one does not restrict the data model to only entities that relate to the informational aspect of objects in the business, but enlarges the data model with entities that will not all be captured in electronic form, e.g., order agreements, there is an indication that one entity model level may be sufficient for an industry-wide application architecture.

The concepts of partner, object, agreement, and transaction that helped us develop and precisely define entities of the business entity model were not helpful in analyzing actual business systems with business users. These concepts were perceived by business users as academic and not pertinent. Certain concepts seem to be beneficial for the development of a system, but not for its use.

The arrival of computer graphics displays has greatly facilitated the task of modeling. Nevertheless we feel that a number of modeling tool features would further improve the understanding of an enterprise model by a business user. For instance, the capability to hide in a flow diagram those processes and flows that do not affect a particular user: A finance person may only be interested in processes and flows relating to finance, or he or she may want to see flows serving the business goal of profit margin increase. Another example is the requirement to show by a click of a mouse the conditions for the execution of a particular process, and the different states which can be reached during its execution. Are we asking too much if we ask for some kind of animation to be applied to flow diagrams? For instance, a business user would be delighted if we could "move" a customer, represented by a symbol, through the flow diagram of the business process Manage Sales, or an order through Manage Product Range. We could stop the customer when he or she picks goods from a shelf, or needs to decide what to do next, and switch over to explaining the entities involved.

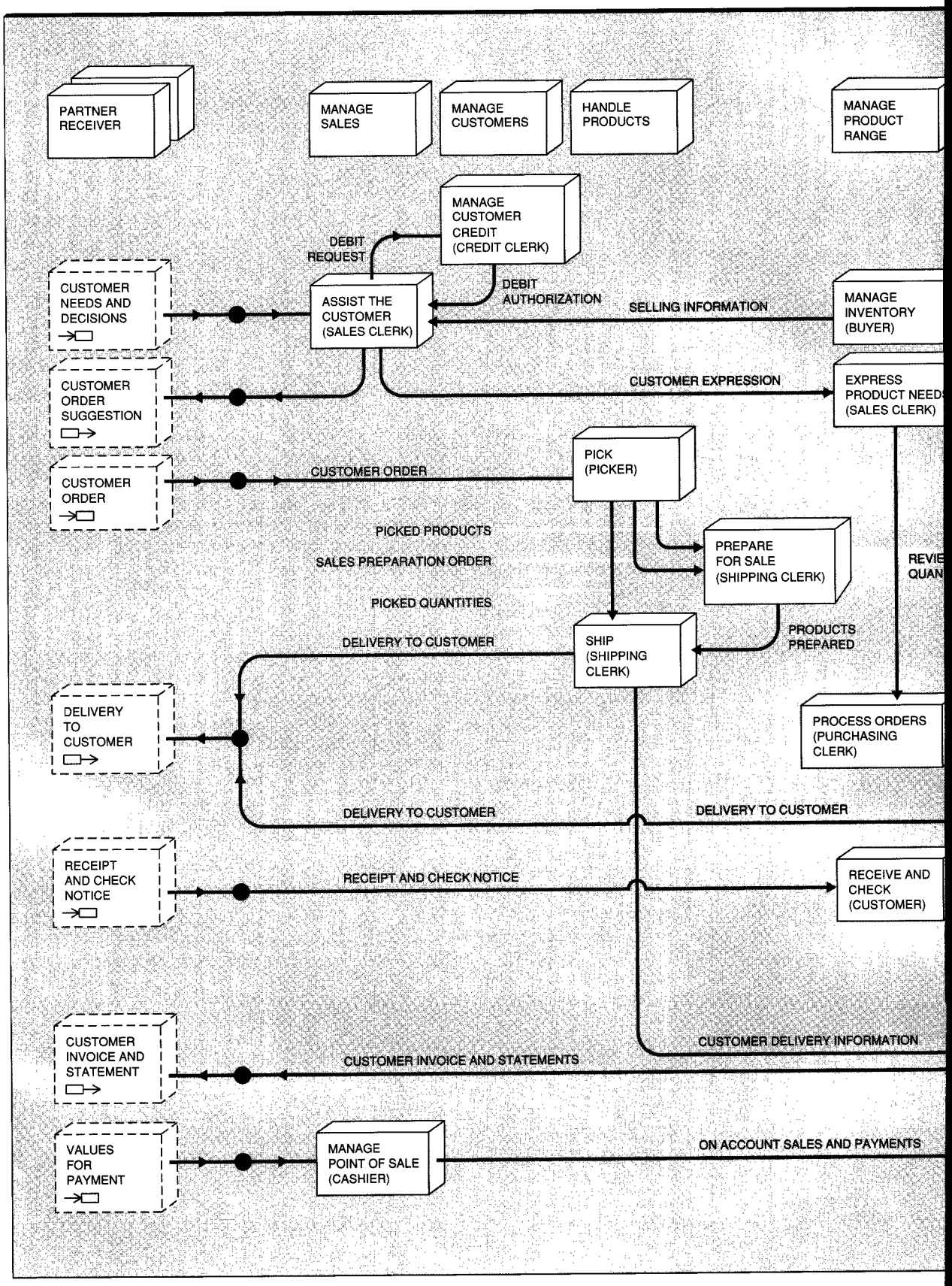
The major next step on which we are currently working is the definition of compliancy rules that will allow us to "tie" applications to the enterprise models of RAA, and the extension of the enterprise models with business concepts such as objectives and goals.

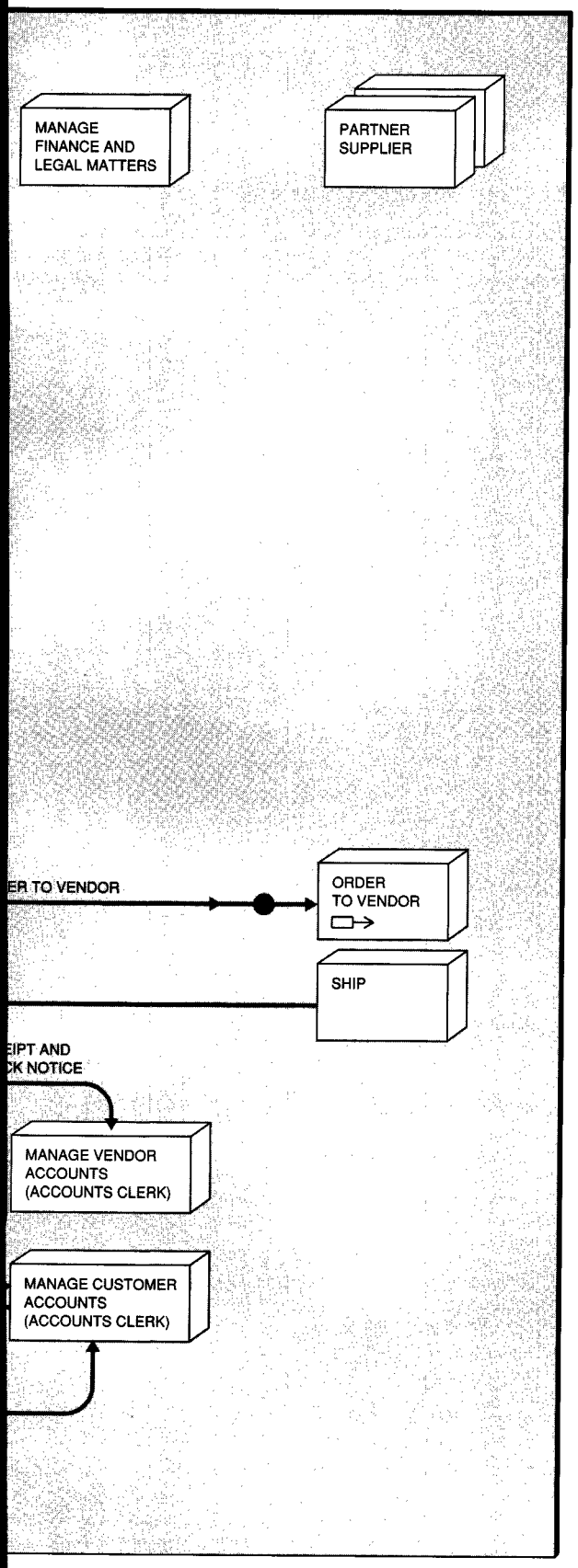
Summary

RAA is a set of offerings by IBM that includes a consultancy service for business analysis, application development, and database development. The core of these offerings consists of the RAA enterprise models made up of a functional model, entity/relationship models, and an organizational model. The particular enterprise modeling approach of RAA consists of a reference architecture on the one hand and a set of workflow and organization examples on the other. By using both, a consultant will be able to develop a customized architecture for a specific retail enterprise with less effort. Another feature of the RAA approach is the role the static functional model plays. Most of the projects of industry-wide application architecture we have studied focus on the data model as the main integrator of applications. The rationale given is that a data model is less prone to changes than a functional model, and that it is even not viable to develop an industry-wide functional model in view of the huge variety of business processes. It is our belief that an industry-wide functional model can be built as well as an industry-wide data model by abstracting from specific cases in an appropriate way.

The challenge of building any industry application architecture is to be general enough to be valid for a wide range of enterprises, but specific enough so that the effort of designing a particular business system or application is limited. We tried to respond to this challenge in three ways: (1) The static functional model describes a generalized business system; the workflow examples that are provided with RAA illustrate how the business processes in the static functional model can be employed to analyze concrete situations such as a customer ordering a customized kitchen, or a buyer selecting a range of garments. (2) We do not define the detailed procedures on how to carry out processes; we leave them to applications that populate the technology model level and tie these applications to the information system model and business model levels through compliancy rules. (3) The enterprise models need to be interpreted

Figure 11 Sales order and delivery





and customized to fit a particular retailer, making the consultant's role so important in RAA.

RAA thus consists of an industry-wide enterprise model, compliancy rules that define the way in which applications implement the enterprise model, and a basic usage process that defines the progression from a reference architecture to a customized architecture and implemented systems (business, application, database). The enterprise model and the compliancy rules together make RAA an application architecture.

Acknowledgments

RAA is the concerted effort of a fair number of people. In particular, the contributions in the area of model concepts, contents, or building by Wolfgang Barth, Nigel Carr, Joseph Cicurel, Karl-Theodor Elig, Joachim Hertel, Keith Mantell, Jean-Marie Riber, John Roskell, Christian Sgard, Arielle Stern-Bony, Harry Stutz, and Colin Thorne are gratefully acknowledged. On the project side, the contributions of Joe Benarie, Paul Grainger, Peter Houghton, Chris Johnson, Steve Leary, Serge Leclerc, Nick Leon, Bob Wilkinson, and Marius Vis are recognized. I am indebted to the anonymous referees whose suggestions improved this paper. Last but not least, I want to thank the "friends of RAA" in Canada, France, Germany, Italy, Spain, Sweden, and Switzerland without whose continuous support RAA would not have been possible.

Appendix: Sales Order and Delivery workflow

The workflow Sales Order and Delivery describes the activities related to sale and delivery by a retailer to a walk-in customer (see Figure 11). A typical example of such practice is a company specializing in kitchen fittings.

Sequence of operations:

1. Customer welcomed and needs outlined.
2. Customer credit checked, and product range available to meet needs determined.
3. Customer needs translated into goods and services on hand, and those that must be ordered from third parties.
4. Customer purchase order with prices printed and signed by customer.
5. Goods on hand picked and prepared for sale.

6. Supplier order issued for third-party goods and services.
7. Goods and services delivered to customer who verifies receipt.
8. Accounting informed and sends invoice to customer. It expects invoice from third-party supplier.
9. Customer pays invoice.

The agent in the organization who carries out the activity is mentioned beneath the business process. Interesting to note is the role the customer plays when acknowledging the receipt and completion of the customized kitchen: The customer acts as an agent for the retailer.

*Trademark or registered trademark of International Business Machines Corporation.

**Trademark or registered trademark of KnowledgeWare, Inc., or the Lego Company.

Cited references and notes

1. *RAA General Information Manual*, GC33-0770, IBM Corporation (1991); available through IBM branch offices.
2. E. F. Wheeler and A. G. Ganek, "Introduction to Systems Application Architecture," *IBM Systems Journal* **27**, No. 3, 250–263 (1988).
3. *Communications Oriented Production Information and Control System, Management Overview,* G320-1230, IBM Corporation (1972); available through IBM branch offices.
4. *Planning Aid for Retail Information System, General Description,* GE20-0582, IBM Corporation (1977); available through IBM branch offices.
5. *Grocery Information Processing System, General Information Manual,* GE20-0702, IBM Corporation (1980); available through IBM branch offices.
6. *CIM Open System Architecture*, ESPRIT Project 688, ESPRIT Consortium AMICE, Brussels, Belgium.
7. P. Stecher, *On the Interface Between Business Systems and Data Processing Systems,* Ph.D. thesis, University of London, England (1978).
8. *Information Processing Architecture*, IBM Corporation, Corporate IS&A, Purchase, NY (1985).
9. J. A. Zachman, "A Framework for Information Systems Architecture," *IBM Systems Journal* **26**, No. 3, 276–292 (1987).
10. P. Stecher, "The Application Controller Concept: A First Experience," *Computer Journal* **24**, 97–106 (1981).
11. J. F. Sowa and J. A. Zachman, "Extending and Formalizing the Framework for Information Systems Architecture," *IBM Systems Journal* **31**, No. 3, 590–616 (1992).
12. J. Martin, *Information Engineering, Book I: Introduction*, Prentice-Hall, Inc., Englewood Cliffs, NJ (1989).
13. H. Tardieu, A. Rochfeld, and R. Colletti, *La methode MERISE*, Les Editions d'Organisation, Paris (1984).
14. Supply phase is different from supply chain, a term frequently encountered in the retail trade. Supply chain consists of the physical movement of goods, starting at a manufacturer, passing through warehouse and store, and ending at a customer.
15. The correct terms for entity and relationship are entity type and relationship type. To simplify reading, we adopt a rather loose wording, as many authors do.
16. The concept of entity-type classes originates in the project, Financial Application Solutions for the '90s, in which major European banks and IBM defined a Financial Services Data Model. The concepts are also described in M. Vetter, *Aufbau betrieblicher Informationssyteme mittels object-orientierter konzeptioneller Datenmodellierung*, Teubner, Stuttgart, Germany (1991).
17. J. Hertel, *Das Konzept der operativen Einheiten in mehrstufigen Warenwirtschaftssystemen*, dissertation, Universitaet des Saarlandes, Saarbruecken, Germany (1991).
18. M. Hammer, "Reengineering Work: Don't Automate, Obliterate," *Harvard Business Review* **68**, No. 4, 104–113 (1990).
19. *AD/Cycle Concepts*, GC26-4531, IBM Corporation (1989); available through IBM branch offices.
20. V. J. Mercurio, B. F. Meyers, A. M. Nisbet, and G. Radin, "AD/Cycle Strategy and Architecture," *IBM Systems Journal* **29**, No. 2, 170–188 (1990).
21. H. M. Sneed, "The Myth of Top-Down Software Development and Its Consequences for Software Maintenance," *IEEE Conference on Software Maintenance*, Miami, FL (1989), pp. 22–29.

Accepted for publication December 1, 1992.

Peter Stecher *IBM Eurocoordination, Market Development, Tour Pascal, Cedex 40, F-92075, Paris La Defense, France (electronic mail: FRIBMS9D@IBMMAIL).* Dr. Stecher joined IBM in 1968 and worked as a systems engineer and product and industry strategist in the manufacturing industry and airlines industry in Germany and England. In 1980 he was assigned to the IBM European Headquarters in Paris to develop an information systems architecture. There he led a Business Systems Planning study for order and inventory business processes. One outcome of the study was a data model. To allow business users to access databases derived from data models, he designed and programmed in Prolog together with Pekka Hellemaa an "intelligent" extraction and aggregation tool for relational databases, a "universal relation interface" called "Smarty." Operational in 1984, Smarty is still in use at IBM European Headquarters. He transferred to Market Development of IBM Europe in 1986 to work on SAA and on strategies for application software, CASE, and computer-assisted software maintenance. From 1989 to mid-1991 he was the architect and development manager of RAA. Dr. Stecher received the degree of Diplom-mathematiker from the University in Munich, Germany, in 1968 and a Ph.D. in computer science from the London School of Economics, England, in 1978. In his doctoral thesis he designed a method and tool for more effective application development, called "Application Controller." The Application Controller can also be regarded as an enterprise modeling methodology and as a real-time knowledge-based system to support the execution of business processes. As the latter it was accepted as an ESPRIT project by the Commission of the European Communities in 1984.

Reprint Order No. G321-5514.