REPRINTED FROM IBM SYSTEMS JOURNAL, VOL26, NO 3, 1987; © 1987, 1999

# A framework for information systems architecture

by J. A. Zachman

***With increasing size and complexity of the implementations of information systems, it is necessary to use some logical construct (or architecture) for defining and controlling the interfaces and the integration of all of the components of the system. This paper defines information systems architecture by creating a descriptive framework from disciplines quite independent of information systems, then by analogy specifies information systems architecture based upon the neutral, objective framework. Also, some preliminary conclusions about the implications of the resultant descriptive framework are drawn. The discussion is limited to architecture and does not include a strategic planning methodology.***

The subject of information systems architecture is beginning to receive considerable attention. The increased scope of design and levels of complexity of information systems implementations are forcing the use of some logical construct (or architecture) for defining and controlling the interfaces and the integration of all of the components of the system. Thirty years ago this issue was not at all significant because the technology itself did not provide for either breadth in scope or depth in complexity in information systems. The inherent limitations of the then-available 4K machines, for example, constrained design and necessitated suboptimal approaches for automating a business.

Current technology is rapidly removing both conceptual and financial constraints. It is not hard to speculate about, if not realize, very large, very complex systems implementations, extending in scope and complexity to encompass an entire enterprise. One can readily delineate the merits of the large, complex, enterprise-oriented approaches. Such systems allow flexibility in managing business changes and coherency in the management of business resources. However, there also is merit in the more traditional, smaller, suboptimal systems design approach. Such systems are relatively economical, quickly implemented, and easier to design and manage.

In either case, since the technology permits "distributing" large amounts of computing facilities in small packages to remote locations, some kind of structure (or architecture) is imperative because decentralization without structure is chaos. Therefore, to keep the business from *dis*integrating, the concept of information systems architecture is becoming less an option and more a necessity for establishing some order and control in the investment of information systems resources. The cost involved and the success of the business depending increasingly on its information systems require a disciplined approach to the management of those systems.

On the assumption that an understanding of information systems architecture is important to the development of a disciplined approach, the question that naturally arises is "What, in fact, is information



 0018-8670/99/$5.00 © 1999 IBM 

systems architecture?" Unfortunately, among the proponents of information systems architecture, there seems to be little consistency in concepts or in specifications of "architecture," to the extent that the words "information systems architecture" are already losing their meaning! Furthermore, it probably is not reasonable to expect reconciliation or commonality of definition to emerge from the professional data processing community itself. The emotional commitment associated with vested interests almost demands a neutral, unbiased, independent source as a prerequisite for any acceptable work in this area.

In any event, it likely will be necessary to develop some kind of framework for rationalizing the various architectural concepts and specifications in order to provide for clarity of professional communication, to allow for improving and integrating development methodologies and tools, and to establish credibility and confidence in the investment of systems resources.

Although information systems architecture is related to strategy, both information strategy and business strategy, this paper deliberately limits itself to architecture and should not be construed as presenting a strategic planning methodology. The development of a business strategy and its linkage to information systems strategies, which ultimately manifest themselves in architectural expression, is an important subject to pursue; but it is quite independent of the subject of this work, which is defining a framework for information systems architecture.

### Derivation of the architectural concept

In searching for an objective, independent basis upon which to develop a framework for information systems architecture, it seems only logical to look to the field of classical architecture itself. In so doing, it is possible to learn from the thousand or so years of experience that have been accumulated in that field. Definition of the deliverables, i.e., the work product, of a classical architect can lead to the specification of analogous information systems architectural products and, in so doing, can help to classify our concepts and specifications.

With this objective in mind, that is, discovering the analogous information systems architectural representations, the following is an examination of the classical architect's deliverables produced in the process of constructing a building.[1]

**Bubble charts.** The first architectural deliverable created by the architect is a conceptual representation, a "bubble chart," which depicts, in gross terms, the size, shape, spatial relationships, and basic intent of the final structure. This bubble chart results from the initial conversations between the architect and prospective owner. A sample of such an initial conversation follows:

"I'd like to build a building."
"What kind of building do you have in mind? Do you plan to sleep in it? Eat in it? Work in it?"
"Well, I'd like to sleep in it."
"Oh, you want to build a house?"
"Yes, I'd like a house."
"How large a house do you have in mind?"
"Well, my lot size is 100 feet by 300 feet."
"Then you want a house about 50 feet by 100 feet?"
"Yes, that's about right."
"How many bedrooms do you need?"
"Well, I have two children, so I'd like three bedrooms."

Note that each question serves to pose a constraint (the lot size) or identify a requirement (the number of bedrooms) in order to establish the "ballpark," or approximate conditions, within which any design will take place. From the above dialogue, the architect can depict what the owner has in mind in the form of a series of "bubbles," each bubble representing a room, its gross size, shape, spatial relationship, etc. (See Figure 1.)

**The architect's drawings are a transcription of the owner's perceptual requirements.**

The architect prepares this bubble chart for two reasons. First, the prospective owner must express what he or she has in mind that will serve as a foundation or basis for the architect's actual design

work. Second, the architect must convince the owner that the owner's desires are understood well enough so that the owner will *pay* for the creative work to follow, and in effect, initiate the project.

Having established a basic understanding with the prospective owner, the architect produces the next set of architectural deliverables, which are called architect's drawings.

**Architect's drawings.** The architect's drawings are a transcription of the owner's perceptual requirements, a depiction of the final product from the *owner's* perspective.

The drawings include horizontal sections (floor plans), vertical sections (cutaways), and pictorial representations depicting the artistic motif of the final structure. The purpose of these drawings is to enable the owner to relate to them and to agree or disagree: "That is exactly what I had in mind!" or "Make the following modifications."

The drawings can be very detailed; however, they are normally developed only to the level of detail required for the prospective owner to understand and approve the design.

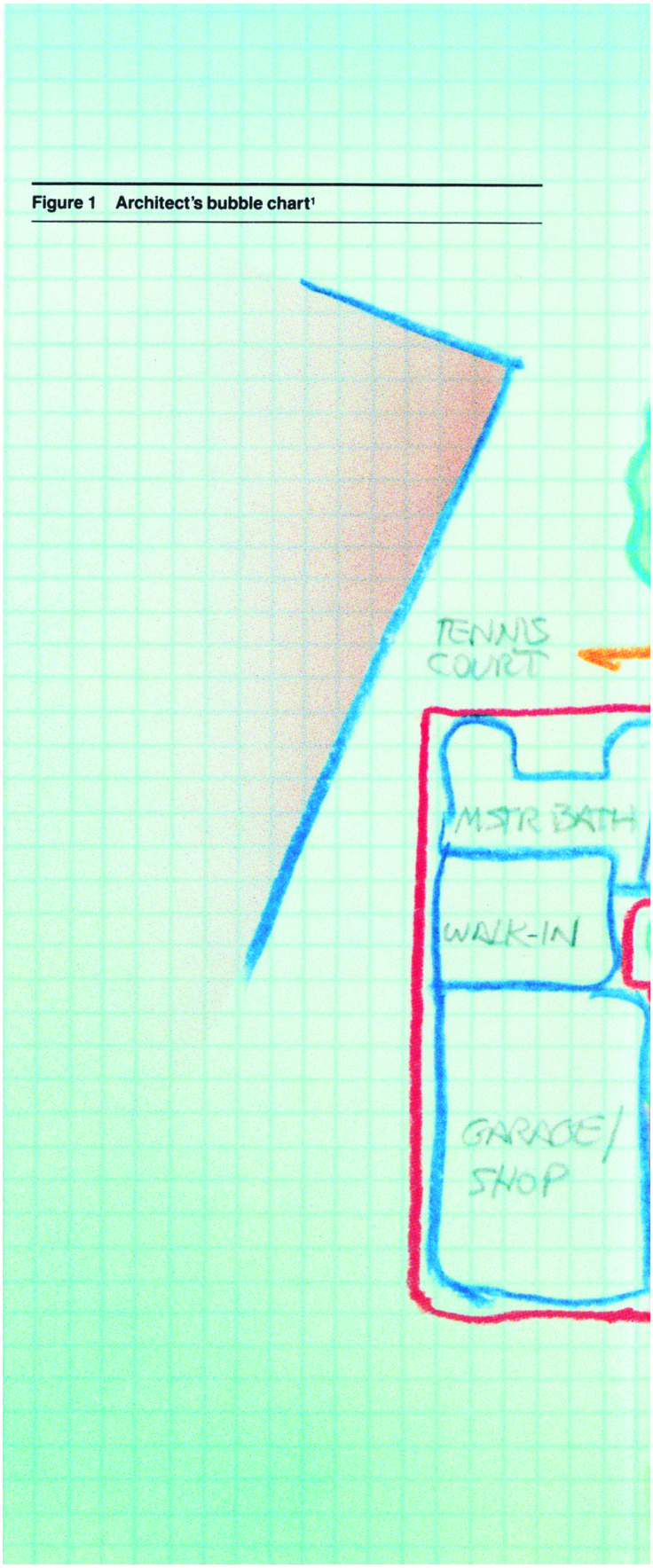
Once the owner agrees that the architect has captured what he or she has in mind, and further agrees to pay the price for continuing the project, the architect produces the next set of architectural deliverables, which are called the architect's plans.

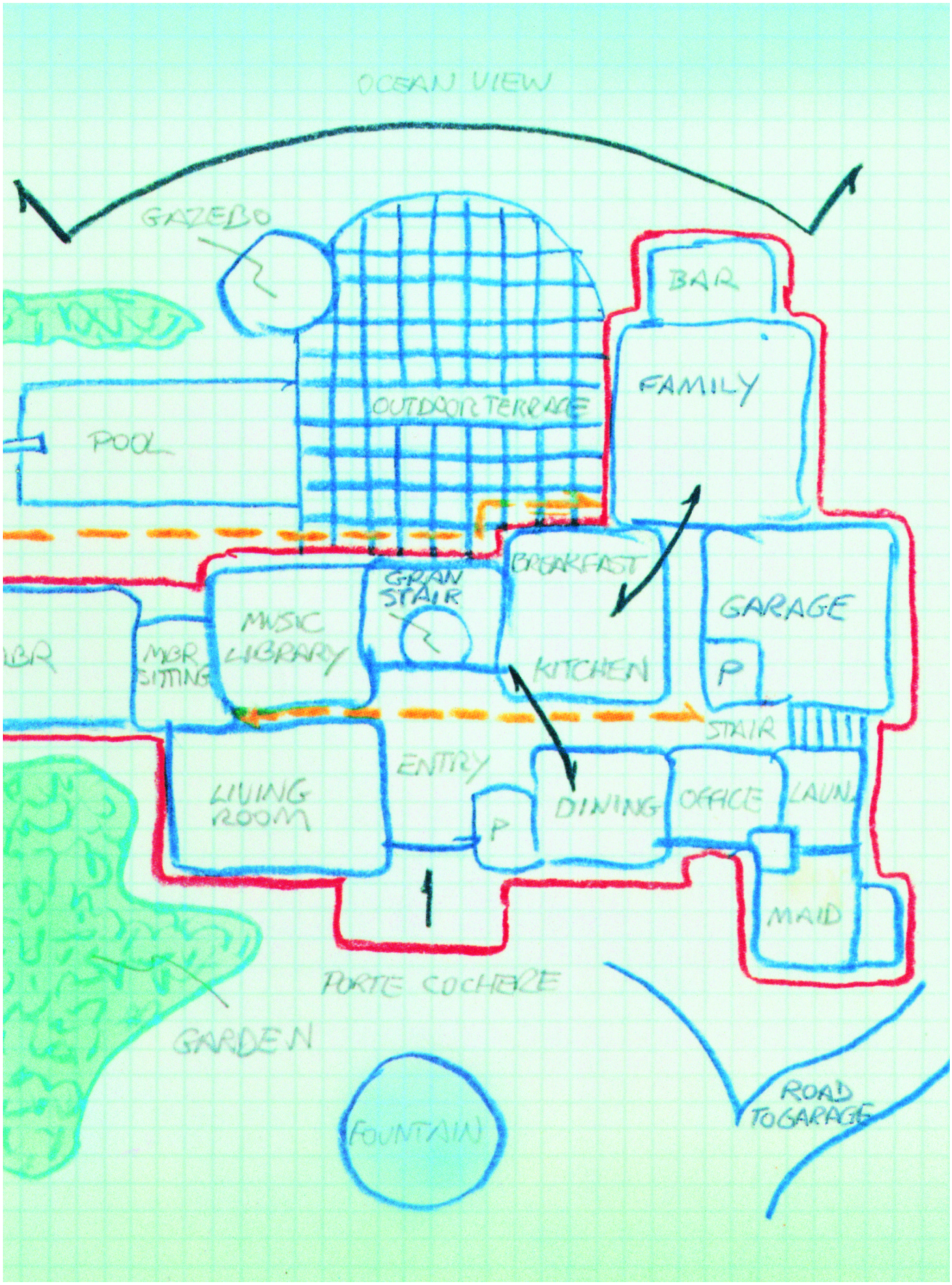
**Architect's plans.** The architect's plans are the translation of the owner's perceptions/requirements into a product. The plans are a *designer's* representation of the final product (as opposed to an *owner's* representation, which is embodied in the *drawings*). The designer's representation (plans) puts an explicit specification around the material composition of the final product.

The plans are composed of 16 categories of detailed representations, including site-work, electrical system, masonry, wood structure, etc. They describe material relationships in the form of diagrams (drawings) as well as bills-of-materials. These plans are the final deliverables prepared by the architect and ultimately become the official "record" of the finished structure.

The architect's plans are prepared to serve as a basis for negotiation with a general contractor. The owner

Figure 1 Architect's bubble chart[1]




OCEAN VIEW
GAZEBO
BAR
FAMILY
OUTDOOR TERRACE
POOL
BREAKFAST
GRAN STAIR
GARAGE
MUSIC LIBRARY
MBR SITTING
KITCHEN
P
STAIR
ENTRY
LIVING ROOM
DINING
OFFICE
LAUN
P
MAID
PORTE COCHERE
GARDEN
FOUNTAIN
ROAD TO GARAGE

takes the plans to a contractor and says, "Build me one of these." If the contractor builds "one of these," which is represented in the architect's plans, the owner knows that there is a high probability of getting the desired product, as depicted in the architect's drawings.

As a result of the negotiations between the owner and general contractor, the plans may be modified because of cost/price and other considerations, but they finally serve to represent what is committed to construction.

**Contractor's plans.** At this point, the contractor redraws the architect's plans to produce the contractor's plans representing the *builder's* perspective. Such plans are prepared because complex engineering products are not normally built in a day. Some phased approach is required which, in the case of a building, may comprise first some site work; next the foundation; then the first floor, and so on, until the building is completed. Furthermore, the contractor may have technology constraints. Either the tool technology or the process technology may constrain his ability to produce precisely what the architect has designed. In either case, the contractor will have to design a reasonable facsimile which can be produced and yet satisfies the requirements. These technology constraints, plus the natural constraints requiring phased construction, are reflected in the contractor's plans, which serve to direct the actual construction activity.

**Shop plans.** Other representations, short of the final structure itself, are prepared by subcontractors. These representations are called shop plans and are drawings of parts or subsections which are an *out-of-context* specification of what actually will be fabricated or assembled. The drawings, architect's plans, and contractor's plans are in-context because the owner, architect, and contractor are all concerned with the entirety of the structure, whereas the subcontractors' representations are concerned with components or parts of the total structure. These shop plans might even serve as patterns for a quantity of identical parts to be fabricated for the project.

**The building.** In the case of producing a building, the final representation is the physical building itself.

In summary, there is a set of "architectural" representations that are produced during the process of constructing a building. The set is given in Table 1.

**Table 1 The set of architectural representations prepared over the process of building a building**

| Representation | Nature/Purpose |
|---|---|
| Bubble charts | Basic concepts for building<br>Gross sizing, shape, spatial relationships<br>Architect/owner mutual understanding<br>Initiate project |
| Architect's drawings | Final building as seen by the owner<br>Floor plans, cutaways, pictures<br>Architect/owner agreement on building<br>Establish contract |
| Architect's plans | Final building as seen by the designer<br>Translation of owner's view into a product<br>Detailed drawings—16 categories<br>Basis for negotiation with general contractor |
| Contractor's plans | Final building as seen by the builder<br>Architect's plans constrained by laws of nature and available technology<br>"How to build it" description<br>Directs construction activities |
| Shop plans | Subcontractor's design of a part/section<br>Detailed stand-alone model<br>Specification of what is to be constructed<br>Pattern |
| Building | Physical building |

## A generic set of architectural representations

Now that we have specified the set of architectural representations produced during the process of constructing a building, it becomes apparent that this set of "architectures" may be generic to the process of building any complex engineering product. A cursory examination of military airframe manufacturing appears to validate this hypothesis as follows:

a. *Concepts* equals *"bubble charts"* (ballpark view). The airframe manufacturers begin with some "concepts," which are specifications for the "ballpark" in which they intend to manufacture. For example, concepts for the final product indicating that it will fly so high, so fast, so far, for such and such purpose, with so many people, etc. are formulated to establish its gross size, shape, and performance.
b. *Work breakdown structure* equals *architect's drawings* (owner's view). The work breakdown

structure is the "owner's perspective." The government requires that the manufacturer specify the work to be accomplished in terms of the components/systems against which costs are accrued and schedules are managed. In this fashion, the government controls the manufacturer in the production of the product.

c. *Engineering design* equals *architect's plans* (designer's view). Engineering, the designer, translates the work breakdown structure into a physical product. The resultant "engineering design" is composed of drawings and bills-of-material.
d. *Manufacturing engineering bill-of-materials* equals *contractor's plans* (builder's view). Manufacturing engineering, the builder, applies the laws of nature and technology constraints to the engineering design to describe how to build the product (i.e., inside-out, bottom-up) and to ensure that everything designed is actually producible.
e. *Assembly and fabrication drawings* equals *shop plans* (detail view). Assembly and fabrication drawings are the instructions to the shop floor personnel on how they are to assemble/fabricate the pieces or parts as stand-alone entities.
f. *Machine tool representation* (machine view). Because manufacturing uses computer-controlled equipment to produce some parts, they insert an additional representation of the final piece or part, short of the physical part itself. This representation is a "program" (i.e., "numerical code program") that is a machine language representation.
g. *Airplane* equals *building* (finished product). The final representation is the actual, physical item itself.

**An analogous set of architectural representations is likely to be produced in building any complex product.**

In any case, there appear to be conceptual equivalents in the manufacturing industry for the architectural representations of the construction industry. This equivalency would strengthen the argument that an analogous set of architectural representations is likely to be produced during the process of building any complex engineering product, including an *information system.*

Before identifying the information systems analogs, it is useful to make some general observations regarding architecture.

First, there appear to be three fundamental architectural representations, one for each "player in the game," that is, the owner, the designer, and the builder. The owner has in mind a product that will serve some purpose. The architect transcribes this perception of a product into the owner's perspective. Next the architect translates this representation into a physical product, the designer's perspective. The builder then applies the constraints of the laws of nature and available technology to make the product producible, which is the builder's perspective.

Preceding these three fundamental representations, a gross representation of size, shape, and scope is created to establish the "ballpark" within which all of the ensuing architectural activities will take place.

Succeeding the three fundamental representations are the detailed, out-of-context representations which technically could be considered architectures because they are representations short of being the final physical product. However, they are somewhat less interesting "architecturally," since they do not depict the final product in total and are more oriented to the actual implementation activities. Nonetheless, they are included in this discussion for the purpose of ensuring a comprehensive framework.

A significant observation regarding these architectural representations is that each 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. Level of detail is an independent variable, varying *within* any one architectural representation. For example, the designer's representation (i.e., architect's plans) is not merely a succeeding, increasing level of detail of the owner's representation (i.e., architect's drawings). It is different in *nature*, in content, in semantics, and so on, representing a different perspective. The level of detail of the designer's representation (i.e., plans) is variable, and quite independent of the level of detail of the owner's representation (i.e., drawings).

**Table 2** The architectural representations produced over the process of building a complex engineering project, along with analogs in the building, airplane, and information systems communities

| Generic | Buildings | Airplanes | Information Systems |
|---|---|---|---|
| Ballpark | Bubble charts | Concepts | Scope/objectives |
| Owner's representation | Architect's drawings | Work breakdown structure | Model of the business (or business description) |
| Designer's representation | Architect's plans | Engineering design/bill-of-materials | Model of the information system (or information system description) |
| Builder's representation | Contractor's plans | Manufacturing engineering design/bill-of-materials | Technology model (or technology-constrained description) |
| Out-of-context representation | Shop plans | Assembly/fabrication drawings | Detailed description |
| Machine language representation | — | Numerical code programs | Machine language description (or object code) |
| Product | Building | Airplane | Information system |

In the same fashion, each of the architectural representations differs from the others in *essence*, not merely in level of detail.

Given this description of the perspectives (i.e., owner's perspective, designer's perspective, builder's perspective, etc.) of architectural representation produced over the process of building a complex engineering product, it is relatively straightforward to identify the analogs in the information systems area, since information systems are also "complex engineering products." Table 2 identifies those information systems analogs along with the building and airplane equivalents.

**Different types of descriptions for the same product.** Before the idea regarding the different perspectives (and therefore the different architectural representations produced over the process of building complex engineering products) is developed further, it is necessary to introduce a second, entirely different idea. Specifically, there exist different types of descriptions oriented to different aspects of the object being described. Table 3 characterizes three such types of descriptions, one of which is oriented to the material of the product, another to its function, and the third to the relative location of its components.

In spite of the fact that each of the descriptions may be describing the same product, each of them is unique and stands alone because each serves quite different purposes. Furthermore, none of the descriptions explicitly says anything about any of the other descriptions. Only assumptions can be made from one about the contents of another. For example, a bill-of-materials exists independently of, and is clearly different from, functional specifications or drawings. Looking at a bill-of-materials tells nothing about functional specifications or drawings (relative locations of components). Only assumptions can be made about function or location, depending upon how descriptively named the parts are in the bill-of-materials. Similarly, the functional specifications say nothing explicit about the bill-of-materials or the drawings, and the drawings say nothing explicit about the bill-of-materials or functional specifications.

In short, each of the different descriptions has been prepared for a different reason, each stands alone, and each is different from the others, even though all the descriptions may pertain to the same object and therefore are inextricably related to one another.

The "description" row of Table 3 suggests that there likely are additional descriptions not characterized in the table as the material description addresses "WHAT," the functional description addresses "HOW," and the location description addresses "WHERE." The implications are that there must be at least "WHO," "WHEN," and "WHY" descriptions as well. Discussion of these additional types of descriptions is reserved for the future, since using only three different descriptions introduces considerable com-

**Table 3 Three different types of descriptions of the same product**

| | Description I | Description II | Description III |
|---|---|---|---|
| Orientation | Material | Function | Location |
| Focus | Structure | Transform | Flow |
| Description | *WHAT* the thing is made of | *HOW* the thing works | *WHERE* the flows (connections) exist |
| Example | Bill-of-materials | Functional specifications | Drawings |
| Descriptive model | Part-relationship-part | Input-process-output | Site-link-site |

**Table 4 Information systems analogs for the different types of descriptions**

| | Description I (material) | Description II (function) | Description III (location) |
|---|---|---|---|
| Information systems analog | Data model | Process model | Network model |
| I/S descriptive model | Entity-relationship-entity | Input-process-output | Node-line-node |

plexity into the subject of information systems architecture at this time. Therefore, the remainder of this paper will be limited to the three types of descriptions contained in Table 3. For future reference, Appendix A is included and contains a preliminary, Table 3-like characterization of the additional descriptive types related to people (who), time (when), and motivation (why).

As was the case with the earlier idea regarding the different perspectives of the different participants in the architecture process, once again it is straightforward to identify the information systems analogs for the elements of the second idea, that is, the different types of descriptions for the same object, as follows:

a. Functional description—In information systems terms this would likely be called a process (or a functional) model, and the descriptive representation would be called the same as the general case, "input-process-output."
b. Material description—Generally speaking, the material description describes the "stuff the thing is made of," which in the case of information systems is data. Therefore, in information systems terms, the analog for the material description would be a data model, and in the data vernacular, "part-relationship-part" would become "entity-relationship-entity." The data model is the equivalent of the bill-of-materials for the information systems product.
c. Location description—In information systems, this would likely be called the network model, in which the focus is on the flows (connections) between the various components. In the information systems network vernacular, "site-link-site" would become "node-line-node."

Therefore, the rows of Table 4, which constitute the analogs in information systems for the more generic types of descriptions, could be added to Table 3.

**The framework.** Two ideas have been discussed thus far:

a. There is a set of architectural representations produced over the process of building a complex engineering product representing the different perspectives of the different participants.
b. The same product can be described, for different purposes, in different ways, resulting in different types of descriptions.

The combination of the two ideas suggests that for every different type of description, there are different perspectives (and actually different representations) for each of the different participants. For example, for the material (or data) description, there are the

owner's representation, the designer's representation, the builder's representation, etc. For the functional (or process) description, there are the owner's representation, the designer's representation, the builder's representation, etc. For the location (or geographic) description, there are also the owner's representation, the designer's representation, the builder's representation, etc.

Figure 2 illustrates the total set of different perspectives for each type of description. Note that because the intent is to depict a framework for *information systems* architecture, all the information systems analog names from Tables 2, 3, and 4 have been used in Figure 2 in place of the more generic manufacturing or construction names. Also, the machine language perspective in Table 2 has been omitted, merely because it is not as interesting as the others from an "architectural" point of view.

The one single factor that makes this framework extremely interesting is the fact that each element on either axis of the matrix is explicitly differentiable from all other elements on that one axis. That is, the model of the business (owner's perspective) is different from the model of the information system (designer's perspective), and so on. (Remember from earlier discussions that these representations are not merely successive levels of increasing detail but are actually *different* representations—different in content, in meaning, in motivation, in use, etc.) Also, the data description column (entity-relationship-entity) is different from the process description column (input-process-output), and so on. Because each of the elements on either axis is explicitly different from the others, it is possible to define *precisely* what belongs in each cell, and further, each cell in the matrix will be explicitly different from all the other cells.

### Architectural representations for describing data

To illustrate how each cell differs from all of the others, examine the data description column of Figure 2. Even though every cell in the column is descriptive type I relating to data, and the descriptive model is "entity-relationship-entity," the meanings of "entity" and "relationship" change with the different perspectives of the participants in the architecture process. The only exception is the scope description (ballpark) cell, in which entity is defined the same as entity in the model of the business cell. This ballpark perspective is merely a very high level of aggregation which is being used like the architect's "bubble charts" to establish the gross size and scope of the data strategy, leading to the decision regarding investment of data processing resources in managing data.

**Scope/description (ballpark perspective)—data column.** The scope description cell in the data description column of Figure 2 could be expected to be a list of all the things that are important to the business, and therefore, that the business manages.[2]

Table 5[3] is an example of an architectural representation in the data description column from the scope description perspective.

This representation would be a list of *things* (i.e., material—grammatically, nouns) as opposed to a list of *actions* (i.e., processes—grammatically, verbs). A list of actions (verbs) could be expected in the next column, process description. The list of things (material) in the data description column would be called "entities" in data vernacular.

Since this architectural representation is at the scope description row, one could also expect that the entities (things) would likely be entity "classes," that is, higher levels of aggregation, because the decision being made as a result of this representation would be one of scope, not one of design. A selection would be being made of the entity class or classes in which to invest actual information system (I/S) resources for data "inventory" management purposes.

Further, in this cell, one might not expect to be definitive about the relationship between entities. The scope decision would constitute overlaying the business values on the total range of possibilities to identify a subset of entity classes for implementation which is consistent with the resources available for investing in information systems—specifically, in this case, the management of the selected class (or classes) of data. Furthermore, it is useful to start with the total list of entities because, at times, the entities that are not selected are as significant as those that are selected.

The strategy/resource investment decision is made by understanding the values/strategies of the business, which can be done by using various data-gathering/analytical techniques. The decision is made by overlaying the analytical conclusions on the total list of business entities in the scope description cell and thereby selecting the subset of business entities in which to actually invest data processing

**Figure 2** **Framework for information systems architecture**

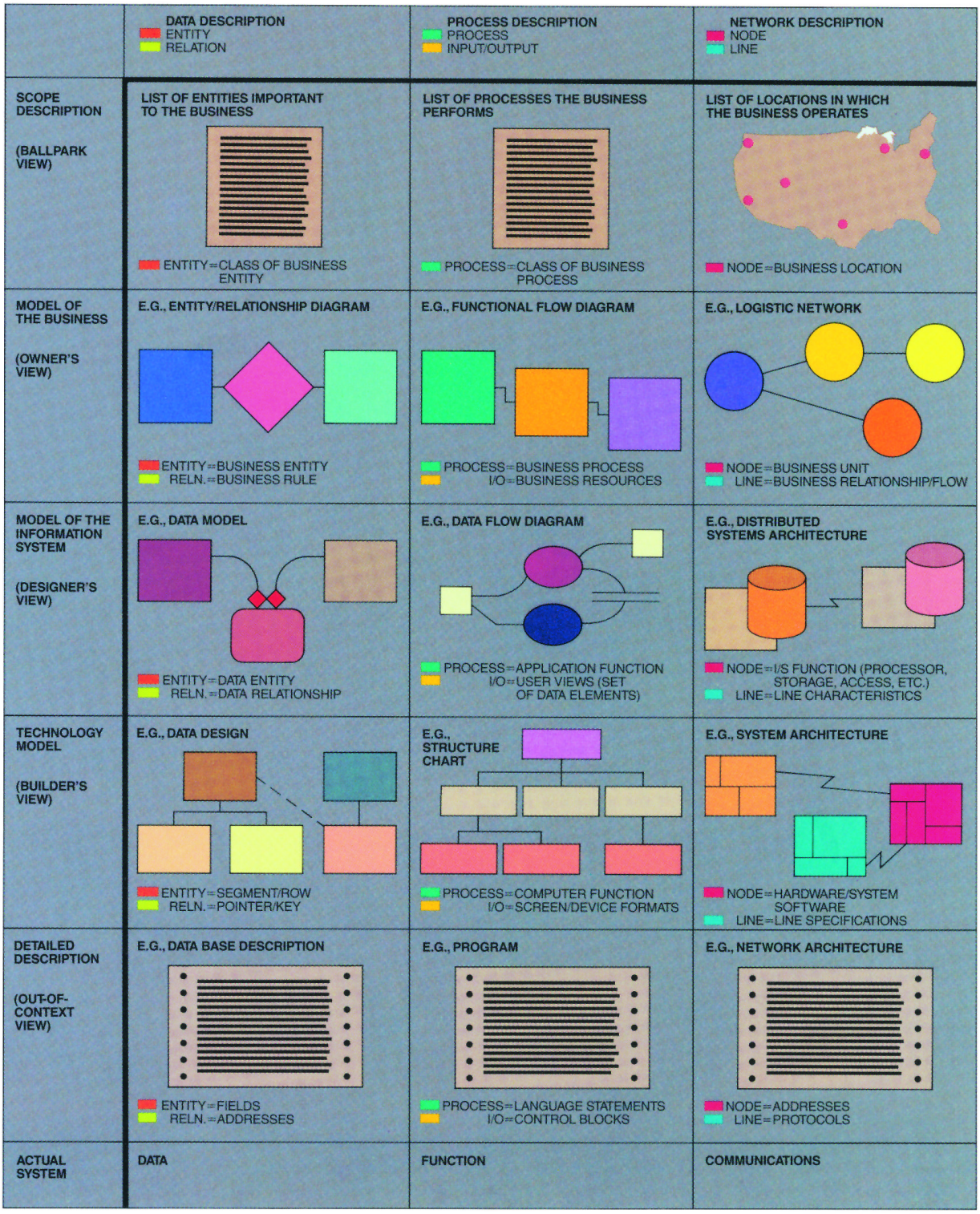


**Table 5 Sample entities—Architectural representation in the data description column from the scope description perspective[3]**

| | |
|---|---|
| Product | Policies and procedures |
| Part | Legal requirements |
| Supplies | G/L accounts |
| Equipment | Accounts payable |
| Employee | Accounts receivable |
| Customer | Long-term debt |
| Supplier | Marketplace |
| Competitor | Promotion |
| Building and real estate | Purchase order |
| Objectives | Customer order |
| Job | Production order |
| Organization unit | Shipment |

resources. Since this paper is intended to define architecture, not to describe strategy methodologies, nothing further will be said about strategic planning except to point out that similar kinds of decisions have to be made relative to every other scope description cell. That is, out of the total list of business processes, the business likely does not have enough data processing resources to automate all the processes. Therefore, a decision will have to be made to select a subset in which to invest data processing resources for actual automation. By the same token, out of the total list of locations in which the business operates, it probably does not have enough data processing resources to put hardware and software in every location. Again, decisions will have to be made in selecting a subset of locations in which to actually install hardware and software.

These are the strategy/resource investment decisions that are supported by the scope description cells in the top row of Figure 2. Although they are inextricably related, the probability is that each decision will have to be addressed independently of the others. Discussion now continues on the framework, particularly the data description column of Figure 2.

**Model of the business (owner's perspective)—data description column.** Since this model (or description) appears in the data description column, the descriptive model will be "entity-relationship-entity," and when owners (users) describe the business and say "entity," what they have in mind are *business* entities.[4]

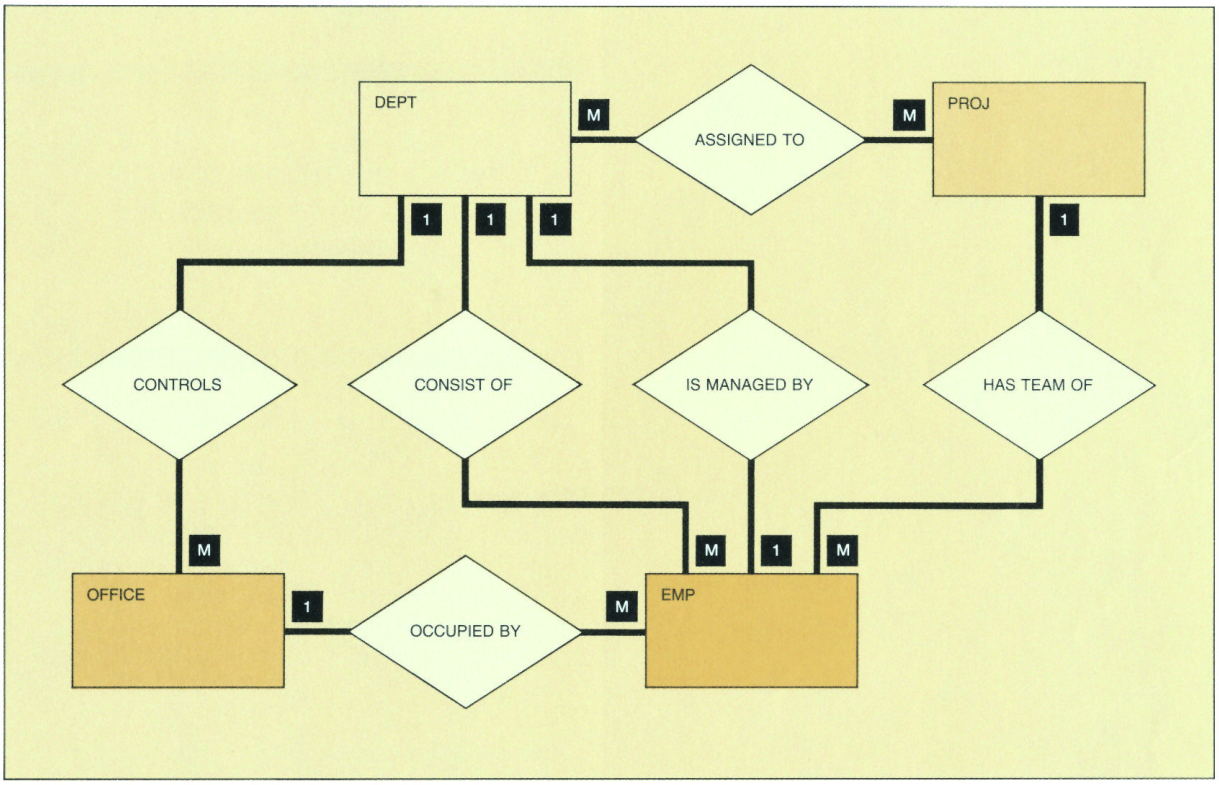
For example, when owners (users) specify an entity such as "employee," what they have in mind would be real beings, that is, flesh and blood employees who work for the business. That meaning of "employee" is entirely different from the one used in an information systems model (the designer's perspective), in which "employee" would refer to a record on a machine, which also happens to be called "employee," however conceptually and entirely different it is. (This *data* entity, as opposed to *business* entity, would be found in the cell directly below.) Figure 3 is an example of a model of a *business*, oriented to data.

Further, when owners, describing a business, specify a relationship between the entities, what they have in mind would be the business rule or strategy that relates one entity to another entity.[5] A business rule or strategy, for example, might be "in this business we ship this product from that warehouse only." An entirely different rule would be "in this business, we ship this product from any warehouse we have." These are business rules and not data relationships such as would be expected in the model of the information system (designer's perspective) in the cell below the Model of the Business shown in Figure 2.

Finding good "real-life" examples which crisply illustrate each of the architectural representations is difficult. There are two reasons for this difficulty. First, when the real-life representations were being developed, no framework existed to clearly define and differentiate one representation from the others. Therefore, many real-life illustrations are a mixture of representations, both conceptually (e.g., business entities and data entities get mixed together) and physically (e.g., entities and inputs/outputs, that is, user views from the process description column of Figure 2, get mixed together). Second, real-life examples are hard to understand because it is not always clear what model, or cell, the author had in mind when developing the representation.

An illustration of this difficulty is found in Figure 3. It is clear that this model is describing *data* and not *process*, but the question is, did the author have in mind a description of a *business* or a description of an *information system*? In this case, it is likely that the description is of a business because of the existence of the "many-to-many" relationships. In real life, there are many "many-to-many" relationships, but the database management concepts that are popular today require that the "many-to-many" relationships be resolved in order to run on a machine. Therefore, "artificial" entities have to be created to resolve the "many-to-many" relationships, and the model in Figure 3 would have to be "normalized"

**Figure 3 Sample entity relationship model — Model of the business (owner's perspective) — Data column[3]**



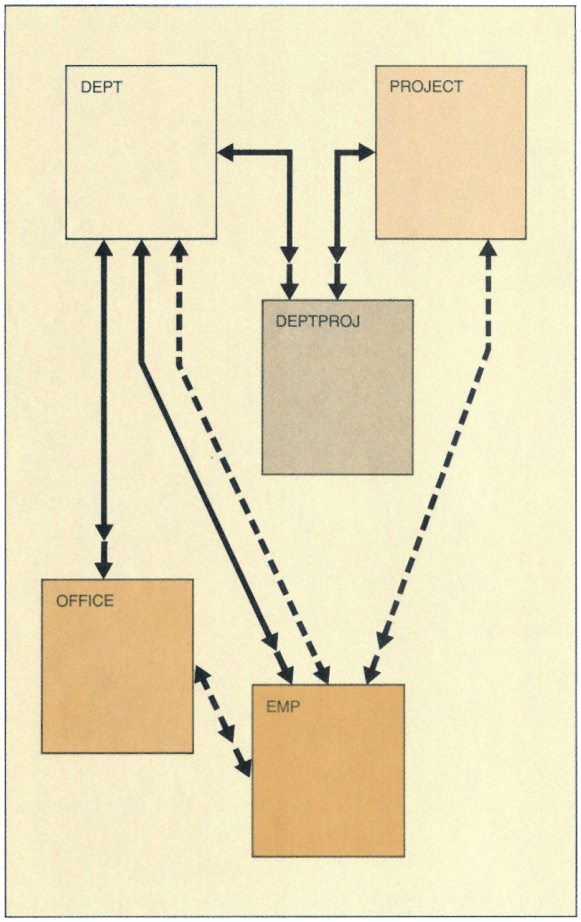

before it could be a legitimate model of an information system. In any case, since the model in Figure 3 is not "normalized," by today's standards at least, it is clearly a model of a business as opposed to a model of an information system.

**Model of an information system (designer's perspective)—data description column.** Once again, since the model of the information system is in the data description column of the framework, the descriptive model used is "entity-relationship-entity." But from the designer's perspective, the meaning of "entity" would change to that of a record on a machine, and relationship would change to that of a data relationship. Clearly, the example in Figure 4 is a model of an information system and not a model of a business, because of the existence of "artificial" entities, specifically the "DEPTPROJ" entity (resulting from the concatenation of department and project), which is not a real-life item, but something in an information system, created in the process of translating the business description into an information systems "product." In the data design vernacular, this example of Figure 4 would likely be called a data model.[6,7]

**Technology model (builder's perspective)—data description column.** As in the previously described cells, the descriptive model in the builder's cell is "entity-relationship-entity," and what could be expected is the physical implementation or data design for the conceptual model of the information system.

In the technology model, the laws of nature and technology constraints are being applied. A decision is made to use Information Management System (IMS) or IBM Database 2 (DB2) or XYZ, and depending on the choice, the meaning of entity and relationship change. In the case of IMS, entity means "segment" and relationship means "pointer." In the case of DB2, entity means "row" and relationship means "key," etc.[5] Figure 5 is an example of an architectural representation of the technology model (builder's perspective) in the data description column of the framework.

**Figure 4** Sample conceptual data model — Model of the information system (designer's perspective) — Data column[3]




**Detailed description (out-of-context perspective)—data description column.** The descriptive model is "entity-relationship-entity" in the detailed description cell. This cell is analogous to the subcontractors' out-of-context descriptions. What could be expected is Data Definition Language. An example might be a DBDGEN in which the entities are specifications of the "fields," and relationships are specifications of the "addresses."[5] An example is shown in Figure 6.

This description is "compiled" to produce the machine language representation (relative addressing—not shown in the figure), which is further "link-edited" to produce the actual physical data (absolute addressing) residing in the machine.

It is clear that real-life examples can be found to illustrate all of the architectural representations for the various viewpoints or perspectives that are created for the data (or material) description of the information system.

**Real-life examples can be found to illustrate all of the architectural representations.**

Although actual samples (figures) for each of the remaining cells are available, no attempt will be made to include them in this paper. Let it be sufficient merely to describe how the meanings of the descriptive model terms change in the remaining two columns as the representations shift from perspective to perspective.

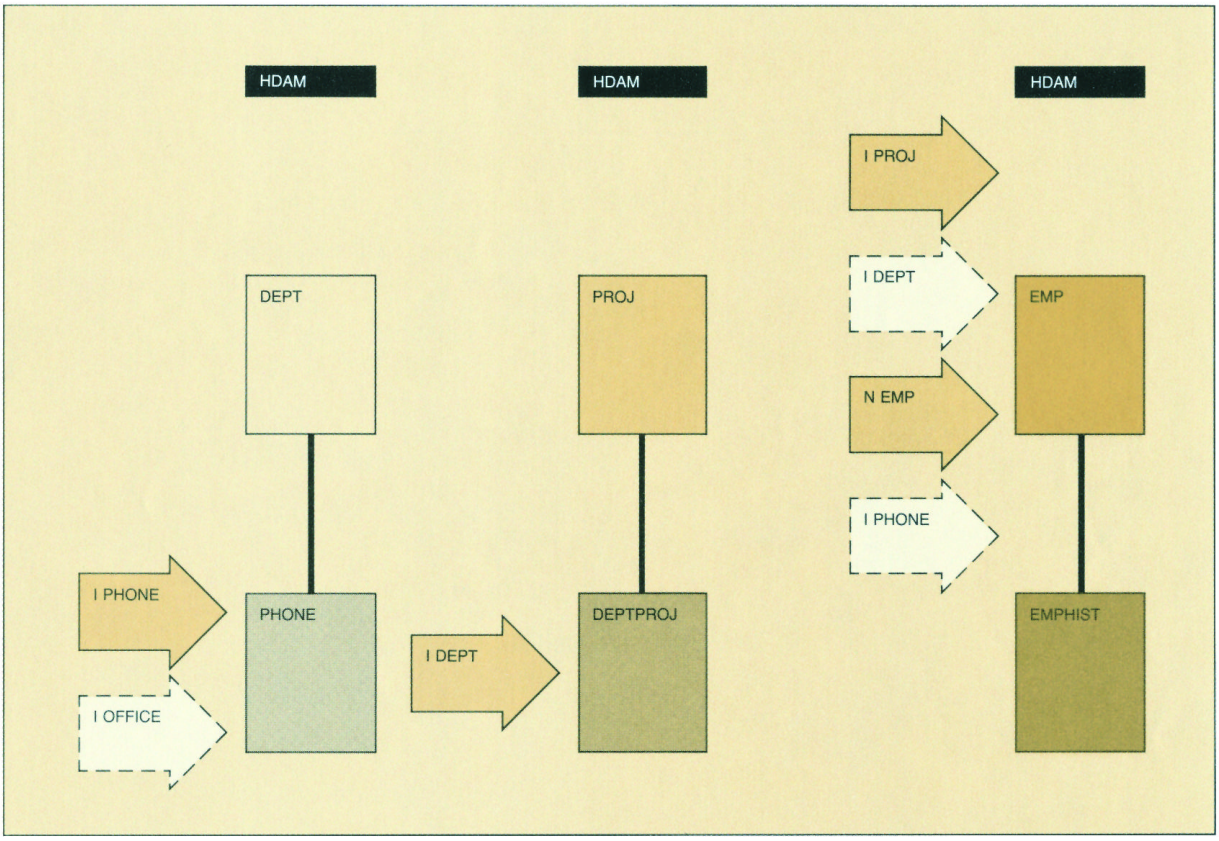
### Architectural representations for describing the process

The descriptive model for describing the process is "input-process-output," and, as in the case of the data description column, each of the representations in the different cells in the process description column of Figure 2 have different meanings associated with "input," "process," and "output."

At the scope description (ballpark perspective) cell, "process" would mean business process. It would likely be some process "class," a relatively high level of aggregation, as the decision being made from the scope description cell is the selection of some subset of the appropriate business processes in which to invest some finite amount of information systems resources for actual automation purposes. Further, in making the scope decision, it is unnecessary to be definitive about the input and output linkages between the functions by overlaying the business values against the total range of automation possibilities. Therefore, just a list of business processes would appropriately be expected in this cell representation.

In the model of the business (owner's perspective) in the process description column, an example might

**Figure 5 Sample DL/I physical model—Technology model (builder's perspective)—Data column[3]**




be a functional flow diagram[8,9] in which "process" would be a *business* process (not an information systems process) and inputs and outputs would be business resources such as people, cash, material, product, etc.

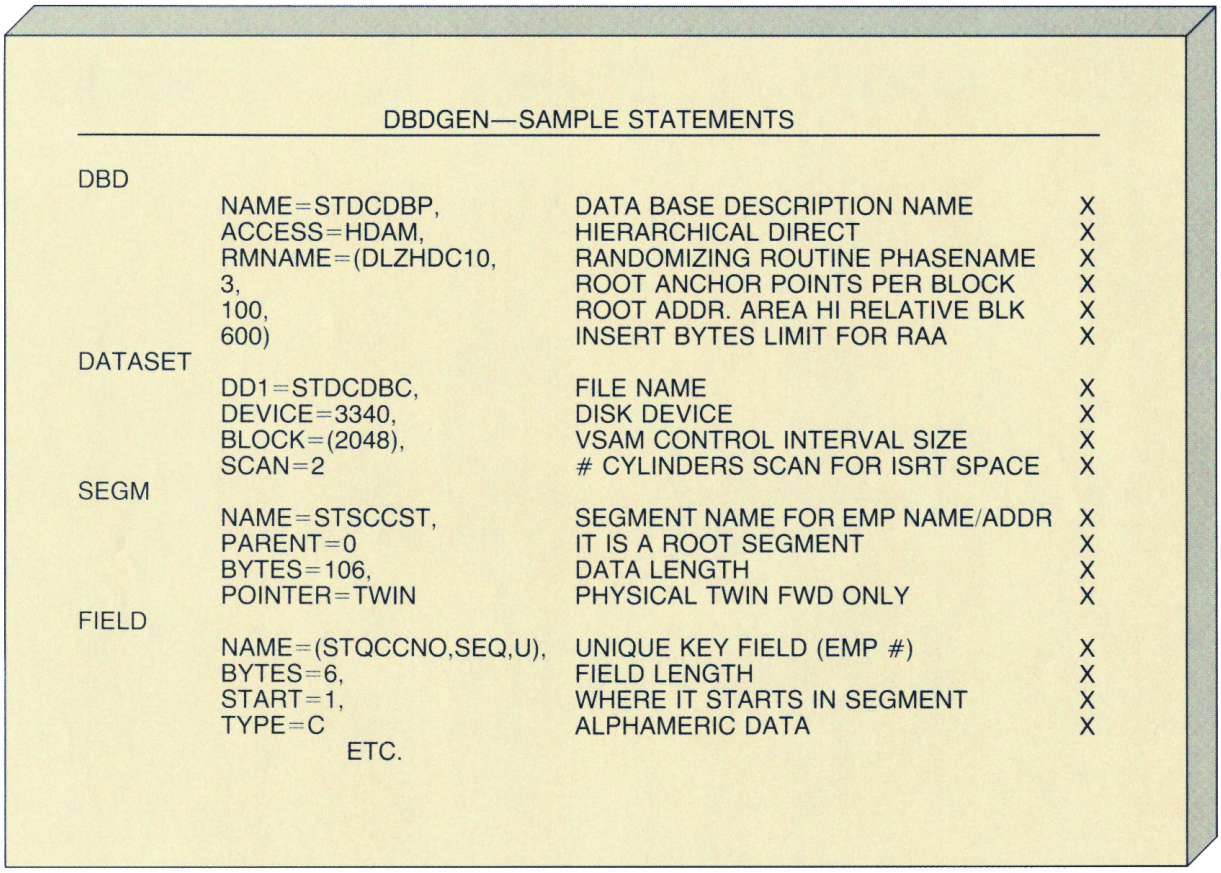
In the model of the information system (designer's perspective) for the process description column, an example would be a data flow diagram[8–10] in which processes are information systems (application) processes (not business processes) and the inputs/outputs are "user views"—some aggregations of data elements that flow into and out of the application processes, connecting them in some sequential fashion. (Note that this does not preclude depicting manual functions that are introduced as part of the information system.)

Proceeding to the technology model (builder's perspective)—process description column, we see that the meaning of "input-process-output" changes once again. In applying the physical constraints of the technology chosen for implementation, for example, IBM 3380 Direct Access Storage Devices versus IBM 3480 Magnetic Tape Subsystems (disks), IMS versus Customer Information Control System (CICS), COBOL versus FORTRAN; IBM 3270 terminal devices (video displays) versus IBM Personal Computers, etc., the builder's perception of a process becomes a computer function, and inputs and outputs are device formats. The predictable example would be a structure chart[10,11] and screen/device formats.

For the detailed description (out-of-context) cell, the example is a "program" in which "process" is a language statement and the inputs and outputs are control blocks.[11,12]

The program is compiled to produce object code, the machine language representation (not shown in

**Figure 6 Sample detailed description (out-of-context perspective) Data column**



DBDGEN—SAMPLE STATEMENTS

```
DBD
         NAME=STDCDBP,          DATA BASE DESCRIPTION NAME          X
         ACCESS=HDAM,           HIERARCHICAL DIRECT                 X
         RMNAME=(DLZHDC10,      RANDOMIZING ROUTINE PHASENAME       X
         3,                     ROOT ANCHOR POINTS PER BLOCK        X
         100,                   ROOT ADDR. AREA HI RELATIVE BLK     X
         600)                   INSERT BYTES LIMIT FOR RAA          X
DATASET
         DD1=STDCDBC,           FILE NAME                           X
         DEVICE=3340,           DISK DEVICE                         X
         BLOCK=(2048),          VSAM CONTROL INTERVAL SIZE          X
         SCAN=2                 # CYLINDERS SCAN FOR ISRT SPACE     X
SEGM
         NAME=STSCCST,          SEGMENT NAME FOR EMP NAME/ADDR      X
         PARENT=0               IT IS A ROOT SEGMENT                X
         BYTES=106,             DATA LENGTH                         X
         POINTER=TWIN           PHYSICAL TWIN FWD ONLY              X
FIELD
         NAME=(STQCCNO,SEQ,U),  UNIQUE KEY FIELD (EMP #)            X
         BYTES=6,               FIELD LENGTH                        X
         START=1,               WHERE IT STARTS IN SEGMENT          X
         TYPE=C                 ALPHAMERIC DATA                     X
                  ETC.
```

Figure 2) which, in turn, is assembled to produce running instructions for the actual, physical system.

Again, it is clear that examples can be found for every descriptive representation for the process description column, as well as for the data description column.

### Architectural representation describing the network

The descriptive model for the network set of architectural representations in the network description column of Figure 2 is "node-line-node."

From the scope description (ballpark) perspective, what could be expected is a list of locations in which the business operates. Therefore, "node" would mean business location, likely at a high level of aggregation, that is, showing little detail about the "contents" of the location. The locations might even be arranged on a map, a geographical construct. If lines were shown, they would probably merely indicate where there are communications or logistics connections between the locations. The purpose served, once again, is the strategy/resource investment decision in which the main decision is to select the subset of locations in which to actually locate technology (hardware/software).

The owner, in describing the business, that is, producing the model of the business as related to the network (or the connectivity characteristics of the business), would perceive the "nodes" to be business units, an aggregation of business resources (people, facilities, responsibilities, etc.) at some geographical location. The lines would represent logistics connections or flows, probably including communications

linkages, but even more basically would represent the distribution structure or logistics network along which communications take place.

In the model of the information system (the designer's perspective) for the network description, the information system designer would perceive the node to be some I/S function, like a processor, storage unit, or access point. This would be a conceptual representation, independent of specific technology

**Technology constraints would be introduced in the technology model.**

which would be introduced in the builder's cell. The line, from a designer's standpoint, would be a communication line at the conceptual level, such as a leased line, dial-up service, U.S. mail, etc. This cell would serve the purpose of making the "distributed systems" decisions, that is, specifying where the I/S facilities would be installed, which of them would be connected, and by what type of connection.

The technology constraints would be introduced in the technology model (the builder's perspective). This cell would depict physical hardware and software, for example, an IBM 3090 processor, 3270 display device, Personal Computer, Multiple Virtual Storage (MVS) operating system, Advanced Communications Function/Network Control Program (ACF/NCP), etc. at the nodes and Synchronous Data Link Control (SDLC), bisynchronous communications, 4800 baud, etc. for the lines.

In the detailed description (out-of-context) cell, the nodes would be addresses, and the lines would be protocols.[13]

In summary, although actual pictures have not been included in the paper, examples could be presented to illustrate every hypothetical architectural representation postulated by the relationship among the various architectural perspectives and the different types of descriptions.

**Table 6 If/then table depicting different architectural representations used within different data processing functions**

| If you are: | Then you probably think architecture is: |
|---|---|
| A programmer | A structure chart |
| The database administrator | Data design |
| An analyst | A data flow diagram |
| A planner | Some combination of entity/relationship diagrams and/or functional flow diagrams |
| The communications manager | The business logistics infrastructure and/or the distributed systems architecture |
| An operations manager | The system architecture |
| A network administrator | The network architecture |
| A program support representative | Detailed data and program descriptions |
| A computer designer | Machine language (not represented on the summary chart, Figure 2) |
| The president | Entity classes, process classes and/or a map |

## Conclusions

When the question is asked, "What is information systems architecture?" the answer is, "There is not *an* information systems architecture, but a *set* of them!" Architecture is relative. What you think architecture is depends on what you are doing. For an example, see Table 6.

We are having difficulties communicating with one another about information systems architecture, because a *set* of architectural representations exists, instead of a *single* architecture. One is not right and another wrong. The architectures are different. They are additive and complementary. There are reasons for electing to expend the resources for developing each architectural representation. And there are risks associated with *not* developing any one of the architectural representations.

Research is being done to create more explicit definitions for each of the architectural representations in this framework, to understand the design issues, the reasons for developing each representation, the risks associated with not developing any one, and the "tool" implications of each cell.

## Summary

In summary, by studying fields of endeavor external to the information systems community, specifically those professions involved in producing complex engineering products (e.g., architecture/construction, manufacturing, etc.), it is possible to hypothesize by analogy a set of architectural representations for information systems.

The resultant "framework for information systems architecture" could prove quite valuable for

- Improving professional communications within the information systems community.
- Understanding the reasons for and risks of not developing any one architectural representation.
- Placing a wide variety of tools and/or methodologies in relation to one another.
- Developing improved approaches (including methodologies and tools) to produce each of the architectural representations, as well as possibly rethinking the nature of the classic "application development process" as we know it today.

## Cited references


1. G. D. Larson, private communication, Gaede and Larson, Architects International Association, Pasadena, CA (1985).
2. *Business Systems Planning—Information Systems Planning Guide,* Application Manual GE20-0627, IBM Corporation; available through IBM branch offices.
3. *Business Systems Planning,* class material, IBM Corporation, Information Systems Management Institute, Los Angeles, CA.
4. D. S. Appleton, "Business rules: The missing link," *Datamation* **30,** No. 16, 145–150 (October 15, 1984).
5. C. J. Date, *An Introduction to Database Systems,* Addison-Wesley Publishing Co., Reading, MA (1981).
6. *ICAM Architecture Part II, Volume V—Information Modeling Manual,* (IDEF1) AFWAL-TR-81-4023, Air Force Wright Aeronautical Labs, Air Force Systems Command, Wright-Patterson Air Force Base, OH 45433 (June 1981).
7. P. P. Chen, *Entity-Relationship Approach to Systems Analysis and Design,* University of California at Los Angeles, Los Angeles, CA (December 1979).
8. T. DeMarco, *Structured Analyses and System Specifications,* Yourdon Press, Inc., New York (1978).
9. S. M. McMenamin and J. F. Palmer, *Essential Systems Analysis,* Yourdon Press, Inc., New York (1984).
10. *HIPO—A Design Aid and Documentation Technique,* GC20-1851, IBM Corporation; available through IBM branch offices.
11. J. K. Hughes and J. I. Michtom, *A Structural Approach to Programming,* Prentice-Hall, Inc., Englewood Cliffs, NJ (1977).
12. H. D. Leeds and G. M. Weinberg, *Computer Programming Fundamentals,* McGraw-Hill Book Co., Inc., New York (1961).
13. *Systems Network Architecture—Technical Overview,* Systems Manual GC30-3073, IBM Corporation; available through IBM branch offices.



**John A. Zachman** *IBM South-West Marketing Division, Western Area, P.O. Box 60737, Los Angeles, California 90060.* Currently the Business Systems Planning consultant for IBM's Western Area, Mr. Zachman joined IBM in 1965 and has held various marketing-related positions in Chicago, New York, and Los Angeles, including one as National Account Manager for a major U.S. oil company. He has been involved with Strategic Information Systems Planning methodologies since 1970 and came to his present position in 1973. He travels worldwide, speaking and consulting on information systems planning, and has written a number of articles on the subject. His current responsibilities include working both internally with IBM and externally with IBM customers in addressing planning approaches to support management with information systems. Mr. Zachman holds a degree in chemistry from Northwestern University. Prior to joining IBM, he served for a number of years as a Line Officer in the U.S. Navy and is a retired Commander in the U.S. Naval Reserve.


## Appendix A: Possible characterization of additional types of descriptions

| | Description IV | Description V | Description VI |
|---|---|---|---|
| Orientation | People | Time | Purpose |
| Focus | Responsibility | Dynamics | Motivation |
| Description | *WHO* is doing what | *WHEN* the events take place | *WHY* choices are made |
| Example | Organization chart | Production schedule | Objectives hierarchy |
| Descriptive model | Organization-reporting-organization | Event-cycle-event | Objective-precedent-objective |