A NEW VIEW OF INFORMATION MODELING

A Bridge Between Data and Information

The information model was conceived to address the complexities of managing large volumes of data, processes, designs, and tools that are shared by many business users with differing requirements. Because an information model derives much of its features from data models, the distinction between information modeling and data modeling is sometimes unclear. One perspective is that information modeling is context dependent: when a model is viewed as a representation scheme for users to comprehend, it is an information model. When used as a representation scheme to be processed by a computer, it is a data model.

Since computers began handling, manipulating, and managing data, the business community has been demanding an increasing degree of sophistication from them. Data base management systems were conceived to integrate the various operations required to process business data. The nature and characterization of data, as well as its structuring and representation, have become quite complex and have given rise to a need to express the logical structure and accessibility of data in a formal way. Data models were developed to represent the data an organization requires.

The business community then found that traditional data and data modeling were too primitive because they were oriented merely toward record keeping and storage. An enhanced notion of data was needed that could capture the meaning of data as well as encapsulate the schema of the data base. To represent this enhanced data, semantic data models were conceived.

As organizations became more complex, so too did their data. Although some semantic data models could eventually describe real-world situations, business data was often too copious for these models to handle directly. Moreover, semantic data models were usually too rigid, and their formalisms lacked the flexibility needed to describe the real world easily. An intermediate model or concept was now needed to capture semantic data generated by complex and non-ideal interactions. Again, from the business community came the request for an information model that depicted the real world in a way readily accessible to users.

The transition from data modeling to information modeling was evolutionary, yet within the business community there is still confusion about the roles and definitions of these terms. There are no specifications for the constructs that can be used by the information model. One school of thought holds that an information model is a glorified version of a semantic data model; another view purports that the information model should be composed merely of English texts combined with graphics. An alternative viewpoint considers that the entity-relationship (ER) model could qualify as an information model, even though the data base community regards the entity-relationship model as a variety of a semantic data model and not an information model. Yet another school of thought regards the way that the schema is captured as the important feature—that is, the information model captures an external schema, whereas the data model captures a conceptual schema.
This article describes the essential points of data modeling and puts forth another view of information modeling in which the information model is context dependent--that is, an information model in one context could be a data model in another.

**DATA MODELING**

Research in data modeling has concentrated in two areas from which two types of models were developed: traditional models and semantic models. Traditional models include the network, the hierarchical, and the relational models. Semantic models include the entity-relationship, the functional, and the object-oriented models. Traditional data models have been used primarily to represent the database, whereas semantic data models have been used to model operational and semantic data.

Recently, a third type of model has been included in the classification of data models. These hyper-semantic data models not only include the constructs provided by semantic data models but provide inference capabilities necessary to model knowledge-based applications. Although semantic data models also provide powerful constructs (e.g., inheritance, generalization, aggregation, and composition), by themselves these are insufficient to model knowledge-based applications, which require inference capabilities.

Currently, the most popular data model is the relational data model that views the world as a set of relations. Exhibit 1 shows information about employees and their departments represented by two relations, EMP and DEPT. EMP has attributes SS#, NAME, SAL, and D# (for the employees' social security numbers, names, salaries, and department numbers); and DEPT has attributes D#, DNAME, and MGR (for the department numbers, names, and managers). Each relation has a primary key that is used to uniquely identify the rows of the tables. In this example, SS# is the primary key of EMP and D# is the primary key of DEPT.

The relational model is popular because of its simple representation. In addition, the relational data model has a well-defined theory. One of the major drawbacks of the relational model, however, is its lack of support for complex objects, which resulted in the development of semantic data models. One such semantic data model that is gaining popularity is the object-oriented data model. The object-oriented data model, unlike the relational model, is not currently embodied in a standard form, although numerous object data models have been proposed.

In an object-oriented data model, the world is viewed as a set of objects that communicate with one another by exchanging messages. Objects that have similar properties are grouped together into a class. Exhibit 2 shows two classes, EMP and DEPT. The objects that belong to the class are referred to as instances of the class. Each object has a unique object ID (or OID). In Exhibit 2, the OIDs are shown inside the circles, which represent the instances. The OIDs need not be the same as the primary key values used to identify the tables in the relational model. A class has properties, called instance variables, that describe the objects of the class.

For example, EMP has instance variables SS#, NAME, SAL, and D#; the class DEPT has instance variables D#, DNAME, and MGR. In addition, classes have methods associated with them. Methods are procedures that are executed when they are invoked. Methods are invoked
when appropriate messages are received. For example, if the salaries of employees have to be increased by $5,000, a message is sent to the class EMP; then a method (e.g., UPDATE(underbar)SALARY) is executed and the value of the salary instance variable of each employee is increased by $5,000.

Two important concepts in the object data model are the IS-A hierarchy and IS-PART-OF hierarchy. The IS-A hierarchy is used to define subclasses (i.e., a class may have subclasses associated with it). Subclasses inherit all the properties of their class. Exhibit 3 shows that the class EMP has SEN-EMP and JUN-EMP as subclasses. This means that all senior and junior employees are employees, and therefore they inherit all the properties of employees. In addition, senior employees have an additional property named Pension Plan, while junior employees have a property named Performance Grade.

The IS-PART-OF hierarchy is used to represent complex objects. For example, a complex object Book has components Cover Page, Set of Sections, and List of References as illustrated in Exhibit 4. An object-oriented model can represent all of the components of a book.

Because information is one of the most critical resources in many organizations, it is crucial and completely as possible so it can be managed properly. Therefore, data modeling will continue to play a key role in organizations in academia, business, and government.

INFORMATION MODELING

The term information modeling was coined by the business community to describe the special way of modeling semantic data within an organization at a fairly high level of abstraction. To represent the vast amount of information that exists in business, and to enable the communication of this information to people with different skills and backgrounds, less formal modeling methods were needed. The amount of formalism that could be left out of an information model was difficult to determine, however. As a result, attempts to characterize information modeling have been subjective and vague. An information model might, for example, be characterized as:

- A variation of a data model.
- A model for the external schema.
- A data model for information retrieval systems.
- Structured English text in the form of templates.
- Graphics.

The diversity of these characterizations has led to confusion. The premise of this article is that from an investigation of the semantics of information models that contrasts their underlying feature against those of data models, a new definition of information modeling emerges in which the information model in one context may be a data model in another. The information model should capture the semantic data of an organization and represent it in a format that is understood by those in the organization, and the data should also be easily captured by the data model and represented in a format that is more amenable to the systems implementation.
To implement these suggestions, a hierarchical layering scheme is used to represent the semantic data present in the real world. This approach provides the framework for developing information and data models for various systems. The layering scheme could be refined when necessary according to the feedback received from the various special-interest groups in the field of data and information modeling as well as from data and information modeling experts.

The layering scheme is illustrated in Exhibit 5. Each layer or level models the semantic data present in the domain of interest (a domain is a slice of the real world). Layer N captures this data at the highest level of abstraction. The formalisms of the models increase as the layers decrease in number. The model used to represent the data at level I (1 < I < N) is the information model for the data models that correspond to layer J (1 < J < I). In other words, when viewed from level I, the model used to represent the semantic data at level J appears to be a data model. But when viewed from level J, the model used to represent the data at level I appears to be an information model.

In some domains, if the format of the information model coincides with that of the data model, then the two models are one and the same. However, for domains that handle large amounts of complex data, the information model and the data model used to capture the semantic data will be different. In this case, the information model is more amenable to the users, whereas the data model is more amenable to the computer system.

The layering scheme illustrated in Exhibit 5 can be regarded as a metamodel for any organization or information system. This metamodel describes the models that are used at the various layers and is not influenced in any way by the layered OSI-ISO model used to represent data exchange in computer networks. In the ISO model, each layer provides a set of services that are built from services provided by the layer directly below it in the hierarchy. This ISO model implies the existence of each layer of the hierarchy.

The metamodel proposed in this article is less strict because it does not use a formal set of criteria to partition the layers. The layers whose models are perceived as being flexible and less formal will be higher up in the metamodel, whereas those layers whose models are perceived as rigid and more formal will be lower down in the hierarchy. As a result, it is not essential that each layer be present in the metamodel. For example, layer M can interface directly with layer J (J < M-1), bypassing layers K and L (J < K < M-1). This is analogous to a programmer bypassing the flowchart drawing phase, even though a flowchart is more abstract than the program code. The intent of the layering approach is to conveniently guide the human thought process through the various levels of abstraction. In reality, no formal boundaries exist in the human mind's conceptualization of notions, ideas, problems, or solutions.

Exhibit 6 illustrates the type of models that could possibly be used at each layer to capture semantic information. For example, ideas are simulated in the representation scheme at the highest layer to capture the semantic data of some domain. This mental model then serves as the information model for the data models used at the lower layers. The modeling process of any domain (i.e., a slice of the real world) does not necessarily have to use the six-layer hierarchy shown in Exhibit 6. Depending on the complexity of the domain, a one- or two-layer hierarchy may suffice. In addition, a layer could use more than one model to represent the semantic data.
For example, the object model together with the ER model could be used at a layer to represent the data.

A layer could be divided further into sublayers. For example, within layer 2, the ER model could be at a higher level of abstraction than the object model. Furthermore, an information or a data model for a complex system could span across layers. In Exhibit 6, an information model composed of the ER model, templates, and pictures spans across three layers. Possible notations for representing these abstract models are illustrated in Exhibit 7. For example, ORLON is a possible candidate for the object model shown in Exhibit 6, whereas INGRES is a possible candidate for the relational model. The IDEF-1X and IRDS data models are variations of the ER model. SQL and Lotus are possible specification languages.

The layering process described can by itself be represented by some data model. Exhibit 8 illustrates such a representation using an object model. The class Metamodel-Class is a class of classes—that is, an instance of this class is also a class. Furthermore, each instance is any layering scheme that conforms to the general scheme proposed in Exhibit 5. An example of an instance of Metamodel-Class is the Abstract-Model-Class. The instances of Abstract-Model-Class are objects. Each such object is an instantiation of the models in Exhibit 6. An example of an instance of Abstract-Model-Class is Representation-Scheme-Object, which is the representation scheme illustrated in Exhibit 7. The instance variables of Representation-Scheme-Object will describe the specific models used at each layer (e.g., INGRES data model, ORION data model, and IDEF-1X).

**BRIDGING THE GAP**

If the intent of the information model is to capture the semantic data in some domain and represent it at a very high level of abstraction, and the intent of the data model is to represent the same data in a rigid formalism that can be processed by the computing system, then it is important that no data should be lost when migrating from one model to the other. There are some possible ways of bridging the gap between the two models.

In the development of an information model and a data model, it is convenient and desirable that they be created independently of each other. At the other extreme, it is prudent (though possibly unrealistic) to expect that their developments should be wholly dependent on each other. Because these conflicting goals can impede the progress of developing a viable combination, it is expedient to settle for the middle ground—that is, the development of the information model and data model can be done in small steps independently.

If development proceeds in this way, then in between each successive step, both models should be combined in order to evaluate whether the semantic data represented in one model could be represented in another without the loss of data. At each stage of the transformation process, appropriate verification and validation techniques should be used to ensure that there is no information loss.

The migration process can be performed by using a series of intermediate models if necessary, with the models organized in a hierarchy of layers. At the highest layer are the models that
comprise the information model; at the lowest layer are the models that comprise the data model. The models at layer I are mapped into models at layer I-1. This mapping is the schema integration process in which the schemas represented by the models at level I are treated as the external schema. The schemas represented by the models at level I-1 are treated as the conceptual schema. The external schemas are then integrated to produce the conceptual schema.

**CONCLUSION**

The term information modeling has sometimes been confused with the related term data modeling. This article has proposed a notional view of information modeling that is based on a hierarchical layering scheme. The boundaries between the layers are vague and based entirely on a user's intuition.

According to this view, however, the information model is context dependent, so that an information model in one context is a data model in another. The information model captures the semantic data of an organization and represents it in a format that is understood by various business users; however, the data is also captured by the data model and represented in a format that can be processed by a computing system.

This article's view of information modeling as it relates to data modeling overcomes the confusion between the meaning of the two terms. In addition, it provides a comprehensive framework with which very large information systems can be developed. Another approach deserving future consideration is one that establishes criteria according to which a model belongs to a particular layer. Although this scheme may be more efficient than the one proposed here, it could also be less flexible and might not be applicable to complex systems. Nevertheless, this view offers the advantage of being able to use prevailing standards.

**EXHIBIT 1 The Relational Model**

<table>
<thead>
<tr>
<th>Relation EMP</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Terry</td>
<td>20K</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>Chris</td>
<td>30K</td>
<td>20</td>
</tr>
<tr>
<td>3</td>
<td>Pat</td>
<td>40K</td>
<td>30</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Relation DEPT</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>Personnel</td>
<td>Ryan</td>
</tr>
<tr>
<td>20</td>
<td>Security</td>
<td>Jamie</td>
</tr>
<tr>
<td>30</td>
<td>Finance</td>
<td>Leslie</td>
</tr>
</tbody>
</table>

**DIAGRAM: EXHIBIT 2 -- The Object Data Model**

**DIAGRAM: EXHIBIT 3 -- The IS-A Hierarchy in an Object Data Model**

**DIAGRAM: EXHIBIT 4 -- IS-PART-OF Hierarchy in an Object Data Model**

**DIAGRAM: EXHIBIT 5 -- Hierarchy of Layers (metamodel)**
Recommended Reading


By Bhavani Thuraisingham and Venkat Venkataraman

BHAVANI THURAISINGHAM is a lead engineer with the MITRE Corp., Bedford MA

VENKAT VENKATARAMAN is a staff engineer with IBM Corp., Rochester MN.

DENIS LEE is a professor of CIS at Suffolk University, Boston

Copyright of Information Systems Management is the property of Taylor & Francis Ltd and its content may not be copied or emailed to multiple sites or posted to a listserv without the copyright holder's express written permission. However, users may print, download, or email articles for individual use.