Common Information Model

James A. Fulton
The Boeing Company, USA (Retired)

INTRODUCTION

A common information model (CIM) defines information that is available for sharing among multiple business processes and the applications that support them. These common definitions are neutral with respect to the processes that produce and use that information, the applications that access the data that express that information, and the technologies in which those applications are implemented. In an architecture based on the CIM, applications map their data only to the CIM and not to other applications, and they interface only to the middleware that implements the CIM (the integration broker or IB), not to other applications. This not only reduces the number of interfaces to be built and maintained, it provides a basis for integrating applications in a way that reduces the coupling among them, thereby allowing them to be upgraded or replaced with minimal functional impact on other applications.

BACKGROUND

The concept of a common information model first emerged in public prominence under the name conceptual schema with the publication of the ANSI/SPARC Database Model (Jardine, 1977). At the time, it was intended as an approach to designing very large shared database systems.

The key thesis of the ANSI/SPARC Committee was that traditional integration practices required point-to-point interfaces between each application and the data sources it depended on, as illustrated in Figure 1. This required the development of a mapping between the definition of the data available from the source (the internal schema) and the definition of the data required by the application (the external schema). The committee referred to such an approach as a two-schema architecture. The interfaces between the components had to implement this mapping in order to transform the physical representation, the syntax, and the semantics of the data from its source to its destination.

Although such computing systems are relatively simple and quick to build, they impose significant problems during maintenance:

- **Number of Interfaces**: As these systems mature, the number of interfaces to be built and maintained could grow with the square of the number of applications. If \( N \) is the number of applications, \( I \) is the number of interfaces, and each application has one interface in each direction with every other application, then

\[
I = 2 \times N \times (N-1) = 2N^2 - 2
\]

This is known as the \( N \)-Squared Problem. Although it is rare that every application talks to every other application, many applications share multiple interfaces. So the equation offers a reasonable approximation.

- **Redundancy**: The transformations required to implement communications are implemented redundantly by multiple interfaces. Each application must design its own approach to merging data from multiple sources.

- **Impact Assessment**: When an application changes in a way that affects its internal or external schema, every application that maps to that schema must be examined, reverified, and possibly revised.

- **Scope of Knowledge**: When an application is being upgraded to support new requirements, or when a new application is added to the architecture, architects have to examine every other application to determine what interfaces are required.

The ANSI/SPARC Committee recommended an alternative architecture, as depicted in Figure 2, in which interaction among applications is mediated through an integration broker that implements what they called a
conceptual schema, and what this paper refers to as a common information model or CIM (also known as a Common Business Information Model, Canonical Business Information Model, Normalized Information Model, Common Business Object Model). In this architecture, individual applications map their schemas only to the conceptual schema and interface only to the component that implements the conceptual schema (herein referred to as the integration broker), which is responsible for translating data from the source application first into the neutral form of the CIM and from that into the form required by the target application. Since each exchange passes from the internal schema of the source application through the conceptual schema of the integration broker to the external schema of the destination, the committee referred to this approach as a three-schema architecture.

Although each communication requires a two-step translation instead of the one-step translation of the two-schema architecture,

- **Number of Interfaces**: The number of interfaces to build and maintain is substantially less, growing linearly with the number of applications, rather than with the square of that number.
- **Redundancy**: The broker manages the common tasks of transforming and merging data from multiple sources, a task that would have to be done redundantly by each application in the two-schema architecture.
- **Impact Assessment**: When an application changes in a way that affects its mapping of its schema to the CIM, the only other mappings that must be examined, reverified, and possibly revised are those that contain data whose definition in the CIM has changed.
- **Scope of Knowledge**: The effect of the three-schema architecture is to hide the sources and targets of data. An application’s integration architects need only know about the common information model; they need not know the sources and targets of its contents. Hence, those sources and targets can, in most cases, be changed without critical impact on the application.

Although the three-schema architecture was published in 1977 (Jardine), it was not extensively implemented for several reasons:

- It increased the scope and complexity of the documentation to be developed and managed in order to keep track of all the schemas and mappings.
- The database technology for which it was intended did not scale well to support multiple applications with different external schemas.
- The change management practices needed to approve a change to the common information model did not meet project schedule requirements.
- Commercial products were being developed that utilized their own embedded databases that did not rely on the conceptual schema.

The end result was that database applications could be delivered faster without using a three-schema architecture, and the recommendation languished.

Today, the common information model has re-emerged as a viable architectural approach, not as an architecture for physical databases but as an approach to achieving application integration. Variations on this approach to achieving the goals of the CIM usually offer one or more of three features:

1. Ontology
2. Standard Exchange Format
3. Integration Framework

**Ontologies**

An *ontology* is a specification of the information appropriate to a particular business process or domain. It defines the entities that can be the subjects of information, the properties of those entities, the relationships among them, and in some cases the fundamental operations that can be performed among them. Like the CIM, an ontology is neutral with respect to applications and technology. Moreover, although it is typically drawn from the vocabulary and practice of a particular community of users, it is also like the CIM, available for use outside that community, and may overlap with the ontologies of other communities. The CIM is essentially an integrated ontology that embraces the ontologies of all the communities that need to share information. A number of standards (described below under Standard Exchange Format) pro-
Related Content

**Multi-Level Modeling of Web Service Compositions with Transactional Properties**
[www.igi-global.com/article/multi-level-modeling-web-service/52991?camid=4v1a](www.igi-global.com/article/multi-level-modeling-web-service/52991?camid=4v1a)

**Document Versioning in Digital Libraries**
[www.igi-global.com/chapter/document-versioning-digital-libraries/11146?camid=4v1a](www.igi-global.com/chapter/document-versioning-digital-libraries/11146?camid=4v1a)

**Main Memory Databases**
[www.igi-global.com/chapter/main-memory-databases/11170?camid=4v1a](www.igi-global.com/chapter/main-memory-databases/11170?camid=4v1a)

**Situational Method Engineering to Support Process-Oriented Information Logistics: Identification of Development Situations**
Tobias Bucher and Barbara Dinter (2012). *Journal of Database Management* (pp. 31-48).
[www.igi-global.com/article/situational-method-engineering-support-process/62031?camid=4v1a](www.igi-global.com/article/situational-method-engineering-support-process/62031?camid=4v1a)