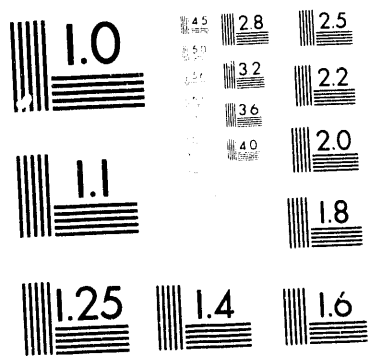
1.0
1.1
1.25
1.4
1.6
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.8

1 of 1

FACT-BASED ENTERPRISE INFORMATION MANAGEMENT USING NIAM

Gary Rivord
Sandia National Laboratories

Introduction

The effort to consolidate the "islands of information" within an enterprise -- to manage information at the *enterprise* level rather than the department or sub-system level -- is known by many names, such as "Information Resource Management" and "Corporate Data Administration". Here we will call it "Enterprise Information Management" (EIM). This effort is becoming increasingly vital as the need for shared information grows, yet it is plagued by costly, time-consuming efforts that produce reams of hard-to-maintain documentation. The results are difficult to integrate, measure, or apply. Management needs a tool that can integrate models from diverse modeling efforts into a global knowledge base, produce metrics to clarify the value of the integration process, and provide a short, traceable route between information models and their physical implementations.

The natural-language emphasis of NIAM makes it an ideal candidate for this tool. When integrated with enterprise-wide data administration, the collection of metrics, CASE tools that produce application code, and automated support tools, NIAM can effectively manage multiple Universes of Discourse (UOD). Prototypes of automated support tools, "Fact Manager" and "Fact Designer", will be discussed.

The "enterprise of interest" used as an example in this paper is an Information Technologies Department at Sandia National Laboratories. This department's mission is to analyze complex application environments and produce information models, physical database designs, and application code. The demand for these services, particularly information models, has increased rapidly in such areas as manufacturing, health care, dismantlement, logistics, and design engineering. The twenty people in this department are implementing their modeling results on a variety of platforms, using both relational and object-oriented database managers.

All papers must include the following statement:
This work performed at Sandia National Laboratories is supported by the U.S. Department of Energy under contract DE-AC04-76DP00789.

1.0 Enterprise Information Management (EIM)

Enterprise Information Management (EIM) is a subject that is much talked about, but is not well defined. EIM has been described as "a means of modeling the major entities and rules which govern an enterprise". This definition is accurate, but incomplete. To complete the definition, the concept of an "enterprise" must be understood from an information perspective.

An enterprise can be viewed as a collection of separate, but related, functional areas. These are known as "Universes of Discourse" (UOD). A UOD includes the real-world objects, facts, constraints, events, and processes relevant to that functional area. Typical UODs in a manufacturing enterprise might include Personnel, Purchasing, Design, Manufacturing, Field Support, Accounting, and Sales.

Prior to the push for enterprise integration, application support for various Universes of Discourse was usually disjoint (Figure 1). Each UOD typically had its own computer system, which was optimized for that application. Information was exchanged between UODs manually or in batch mode using documents, tapes, forms, etc. Information queries spanning more than one UOD were usually impossible. Even if physical networks existed between the UODs, consistent definition and understanding (i.e. logical integration) of the information did not. For example, one UOD might manage PARTS identified by PART-ID, another UOD might manage COMPONENTS identified by COMPONENT-NAME. Is PART the same real-world object as COMPONENT? If so, is PART-ID the same identifier as COMPONENT-NAME? Do PART and COMPONENT share common attributes? Such questions could often be answered by a human "expert" familiar with both UODs, but the power of the computer to select, refine, and present the information was lost.

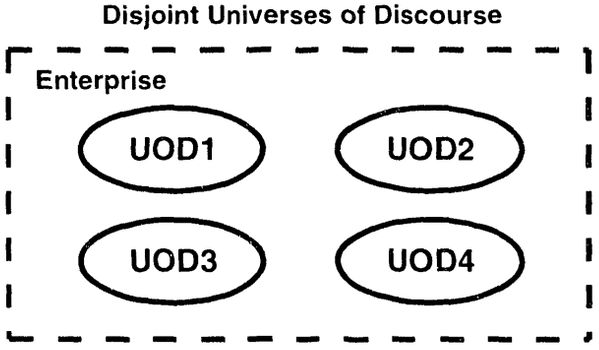

Figure 1

Since no functional area of an enterprise really stands alone, the information used in one UOD is often duplicated in others. Lack of logical integration prevents a full understanding of the extent of the duplication. Figure 2 depicts shared information and interdependencies among the UODs for a given enterprise. Note that a UOD can be totally contained within another UOD, thus creating a supertype/subtype relationship between UODs. The enterprise itself is a UOD. The "sub-UODs" within an enterprise comprise a complete functional breakdown of the enterprise.

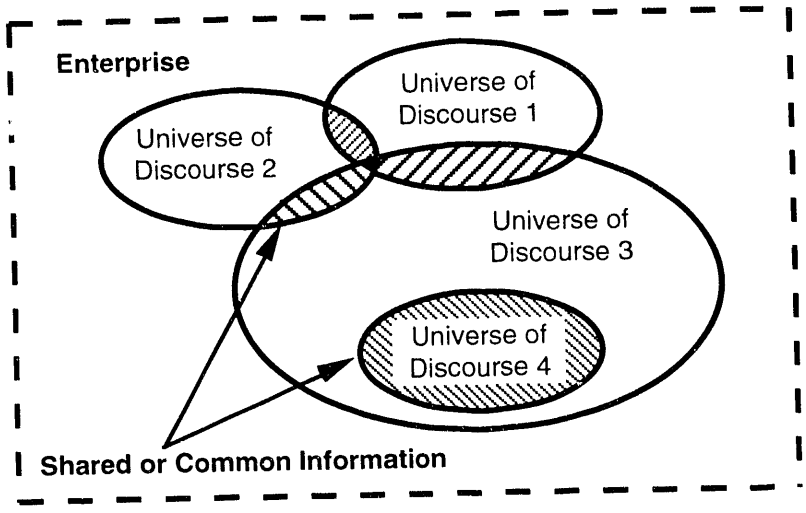

Figure 2

In information terms, each UOD can be broken down as follows: A UOD is made up of one or more real-world objects (or "entities"), each of which plays roles in the UOD just as the UOD itself plays a role in the enterprise. "Elementary facts" are assertions about the roles that a given object may play in relation to other objects within the UOD. Constraints exist concerning when an object may come into being, and what roles it may or must play. Events take place within the UOD, possibly triggering processes that affect objects and their relationships. A UOD can be described as the sum of its objects, elementary facts about relationships between objects (including roles), constraints, events, triggers, and processes. These terms are defined as follows:

Definition of Terms

The term "object" is much overused. "Objects" as used here are the real-world "things" within an enterprise about which information is to be kept, i.e., which are to be managed within the formal information systems of an enterprise. For this discussion, object is equivalent to the concept of entity in the Entity-Rleationship paradigm. An object may be tangible (PERSON, PART) or abstract (PROJECT TASK, PART STATUS HISTORY). This definition of the term "object" varies from that used by "Object Oriented" database management systems, i.e., this definition refers to "conceptual" real-world objects, without regard to their representations in a physical database.

The term "elementary fact" describes a single relationship between objects. Elementary facts cannot be split into smaller statements without having the information expressed by them get lost. Examples of elementary facts are, "*a part is the design responsibility of an engineer*" and "*a part may be used on many assemblies*". It is important to understand the difference between elementary fact types and elementary fact instances. Fact types describe the meaning of the fact in a general sense and is intended to be valid for all instances (specific cases) of the fact type. For example : *"Part is the design responsibility of an engineer"* is a fact type describing the relationship between parts and engineers. *"PART with part number XX2234 is the design responsibility of engineer with employee number EE116758"* is a specific example of the previous fact type. It expresses a specific instance of the relationship expressed by the fact type.

A "constraint" defines the rules governing the "population" of facts. For example, if a rule exists that "*a part must have a responsible design engineer*", then a part may not be registered in the database without "populating" its relationship to design engineer.

An "event" is a real-world occurrence within an enterprise which is represented in an information system by a change to one or more fact instances. Events of interest in enterprise modeling involve only those events which have a possibility to change information (other than the fact associated with the event) in the enterprise's information base. For example, "*person terminates employment*" is an event which might be recorded as a change in the value of the person's "on-roll status" and may cause change to the information base if certain conditions are met.

A "trigger" describes the conditions which when true, indicate that a given process is to be initiated as the result of an event occurrence. For example, on the event that a person should terminate employment, the following trigger would be processed:
On person terminates employment,
if person is responsible design engineer for a part,
Then start process reassign-design-responsibility".

The general form of triggers can be expressed as:

On **<event>**
If **<condition>** is true
Then start **<process>**

It is important to note that a direct relationship between events, conditions, and processes and facts defined in the information base exists and must be maintained.

A "process" is defined in terms of a series of steps which create, retreive, update, or delete information within a UOD. For example, the process "*reassign design responsibility*" might consist of the following steps:

Step 1	Retreive the fact : Design Engineer is responsible for the design of Part where Design Engineer is the terminated employee.
Step 2	Retreive fact : Person is manager of Employee where Employee identifier is that of the terminated employee.
Step 3	Update fact : Design Engineer is responsible fot the design of Part with manager identifier and part identifier.

In a nutshell, Enterprise Information Management (EIM) can be described as:

1. Identifying all of the "objects of interest" of an enterprise within an integrated knowledge base.

2. Knowing all of the relationships (facts) between objects, and the roles each object plays in each relationship.

3. Locating which objects and relationships exist in each Universe of Discourse (this provides the mapping necessary to conduct integration).

4. Identifying all of the constraints which govern the relationships.

5. Identifying all of the events (real world and software-initiated) in the enterprise.

6. Understanding what conditions define each trigger.and which processes are triggered by which events.

7. Defining each process in terms of a series of steps that select, insert, modify, or delete object information within each application.

8. Assigning "ownership" for each of the objects, facts, constraints, events, triggers, and processes, so that responsible individuals may exercise direct control over the model.

2.0 Problems with current tools

Enterprise Information Management (EIM) must be a dynamic, iterative process, and must involve those who actually use the information. Most current business modeling methodologies suffer from an inability to cope with change, and are generally inaccessible to those really responsible for the information.

Typically, an intensive effort is initiated to "model the business", pulling together the best people from each functional area. This elite team is provided with tools they barely understand, and asked to create a model of the enterprise in a short period of time. The effort tends to run over budget and past schedule, and to fall considerably short of providing the desired long term benefits. Documentation is produced in large quantities, but quickly becomes out-of-date, and seldom is referenced in the future. The model produced is usually a "snapshot" of the enterprise at a particular point in time. The enterprise may have changed before the initial model is completed. The model is represented in the form of technical diagrams such as entity/relationship charts, process flow diagrams, flow charts, and functional decomposition charts. These types of representations are readable by experts who understand the syntax, but are of little value to people who are not trained to interpret them. This situation creates a serious problem in the validation of the model, both in terms of accuracy and completeness. It is preferable to have the model reviewed and validated by all areas in the enterprise: from the shop floor to the executive office. Information must be represented such that a diverse audience can accurately interpret it with minimum training. Valuable knowledge is hidden in complex diagrams and large volumes of text.

To cope with change, the output from EIM must change as fluidly as the enterprise changes. The methods used must be integrated into the iterative process of information systems development . The representations produced must be able to support "what if" analyses to evaluate the impact of proposed changes to the information architecture. A well-defined path must exist to disseminate changes to the enterprise model into implemented applications. This path must include interfaces commercial, off the shelf (COTS) software engineering and design tools. The enterprise model is a super set of the knowledge contained in the collective set of design tools in the enterprise.

To be understandable to end users, the rules, processes, and behavior of the enterprise should be represented in a language which can be understood without special tools or training. What better representation than the language that the end users themselves use? The NIAM technique provides the means to structure information requirements into a form that is rigorous enough for implementation *without* losing its comprehensibility to end users.

3.0 Data Administration

The steps involved in the NIAM Conceptual Schema Design Procedure (CSDP) are well documented (Professors G. M. Nijssen, University of Queensland and E. D. Falkenberg, Katholieke Universiteit, Nijmegen). A brief outline of the NIAM CSDP as described in "Conceptual Schema and Relational Database Design; a fact oriented approach, " G.M. Nijssen and T.A. Halpin, is shown.

CSDP Steps

Step 1	Transform familiar information examples into elementary facts.
Step 2	Create a first draft of the conceptual schema diagram and apply population checks.
Step 3	Trim the schema and identify derived fact types.
Step 4	Add uniqueness constraints for each fact type.
Step 5	Identify all splittable fact types (determine proper "arity").
Step 6	Add entity types, mandatory roles, subtypes and occurrence frequency constraints.
Step 7	Check that each entity can be uniquely identified.
Step 8	Add equality, exclusion, subset and other constraints.
Step 9	Check that the conceptual schema is consistent with the original examples, has no redundancy, and is complete.

CSDP has been used effectively to develop high quality information models for specific UODs. But can it effectively provide logical integration *across* UODs? Figure 3 depicts the non-integrated development of models for different UODs within an enterprise. In this diagram, high quality models might be created for each UOD, but with little or no sharing or reuse of model information. It would be pure luck if the models were logically consistent in their definition and naming of the objects and the relationships between them. Clearly, what is needed is a more effective technique to facilitate model development and to manage the models once they are completed. The enterprise requires a formal process integrated with the CSDP which will provide the control necessary for enterprise model management.

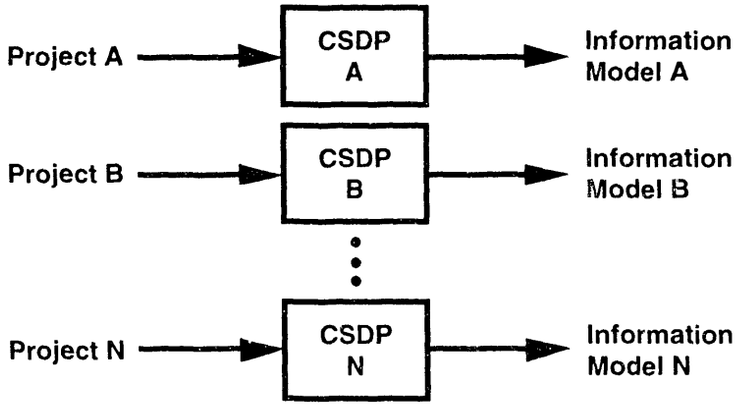

Figure 3

4.0 Data Administration Goals

Like any administrative function, effective data administration must have clear goals, and metrics to measure progress towards those goals. Metrics both keep the effort on track and quantify its value to the enterprise. Suggested goals for data administration, and their metrics, are listed below.

Goal 1 Improve the quality of information modeling and documentation.

A formal information model report format is adopted and used to represent the deliverables for all information projects within the enterprise. The report includes natural language expressions of facts and constraints, objects lists, examples, and NIAM diagrams (Appendix A). The measure of its effectiveness is customer feedback on its understandability and validity.

Goal 2 Maximize the reuse of registered facts.

All facts are registered, meaning that they have been reviewed and approved jointly by the data administrator the information modeler and the fact users and owners. The registration process (discussed in more detail later in this paper) assures fact completeness and accuracy. Registered facts are stored in a fact base. Modelers in new functional areas may select registered facts from the fact base for use. This allows them to concentrate on identifying new facts, instead of redefining existing ones. Reuse naturally promotes logical integration across diverse UODs. The measure of effectiveness of registered facts is the increase in fact reuse in each new modeling exercise.

Goal 3 Minimize the number of objects to be managed.

Standard names are provided for objects, and serve as the preferred names as the objects are reused in other UODs. Aliases are assigned to objects in those cases where a specific UOD cannot use the standard name. Object aliases are stored as facts about the objects in the fact base. The measure of the effectiveness of standard object names is the diminishing rate of new object names as new applications are developed.

Goal 4 Improve the quality of all information models in the enterprise.

This is the most important goal. The goal is for new information models to be right the first time and every time, meaning that all requirements expressed by users are accounted for in the model and implemented in the application(s). The process guarantees this whether the modeler is a well-seasoned or a first time novice. Techniques used include:

* A global knowledge base of registered objects and facts is provided to all modelers.
* Data administration is integrated into the process.
* 100% traceabilitiy is provided from the information model to the physical database structures.
* 100% traceability is provided from application screens and reports to the information model.

Measures of effectiveness of process quality include the percentage of developers using the global knowledge base, the number of times models are iterated before final approval, the percentage of fields found on screens and reports which can be traced to registered facts, and the percentage of database records or fields which can be traced to registered facts, and subjective user assessments concerning the ability to move between applications.

Goal 5 Reduce modeling cost, shorten schedules.

The combination of effective documentation, object and fact reuse, and well-defined tools and processes results in higher quality models at lower cost and in a shorter time. Measures of effectiveness include staffing levels per project (persons per fact), shorter time from initial requirements to final model (hours per fact), and lower development costs (dollars per fact).

5.0 EIM Process

Figure 4 shows the integration of fact registration and reuse with CSDP. Registration and reuse of facts yields the logical integration of models from each of the UODs pictured. "Fact Registration" and "Fact Reuse" are formally defined processes, subject to quality improvement. Fact Registration is the process of populating the fact base with new knowledge. Fact Reuse is the process of selecting registered facts from the corporate fact base for new applications. Both processes are built around the global knowledge base as shown in figure 4, and are "owned" by data administration.

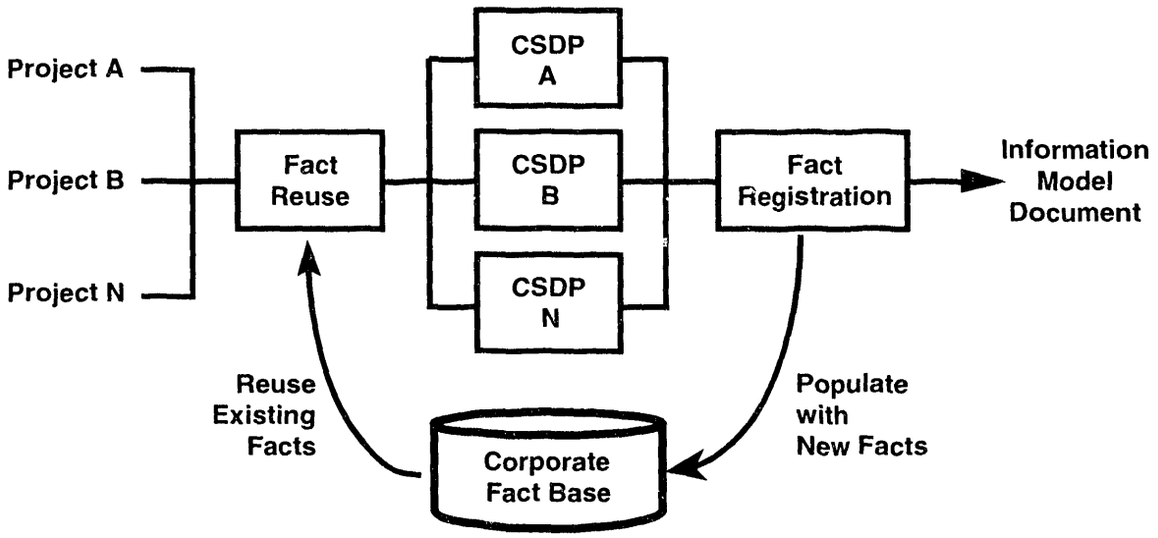

Figure 4

5.1 The Fact Registration Process

The primary function of data administration is the population and management of the fact base. The corporate fact base represents the collective content of all information models within the enterprise. Figure 4 demonstrates the process of populating the factbase - Fact Registration.

Steps performed by the data administrator in fact registration include:

1. Check the completeness of the information model - the model must be complete in terms of object referenceability. Are all objects identifiable? Are there adequate examples? Are the references to owners, source documents, and related projects complete? Does the model conform to NIAM representation standards?

2. Register new objects. Each object is assigned a unique object number. At least three examples of the object identifier are kept on file. Examples of the object identifier are examined to determine if the new object is an alias of any other registered object.

3. Register new facts. A unique number is assigned to each new fact. Constraints are supported by

concrete examples. Facts are associated with one or more UOD.

4. Approve the model. The data administrator and the modelers approve (sign) the model after thorough review and registration.

5.2 Constraint Issues with Multiple UODs

Current NIAM does not clearly describe how facts participate in an environment of multiple UODs such as those found in a typical enterprise. Fact constraints may differ depending on the UOD in which the fact is used. Consider the following example:

Fact: "PERSON is team member on a PROJECT"

Examples of this fact for UOD 1 might include:

PERSON with PERSON-ID 101 is team member on PROJECT with PROJECT-ID P1
PERSON with PERSON-ID 101 is team member on PROJECT with PROJECT-ID P2
PERSON with PERSON-ID 221 is team member on PROJECT with PROJECT-ID P1
PERSON with PERSON-ID 223 is team member on PROJECT with PROJECT-ID null
PERSON with PERSON-ID null is team member on PROJECT with PROJECT-ID P3

The following constraints can be derived from this example set for UOD 1:

A PERSON can work on many PROJECTS.
A PROJECT may include many PEOPLE as team members.
A PERSON does not have to be a team member on any PROJECTS.
A PROJECT does not have to have PERSONS as team members.

The same fact for UOD 2 might have the following examples:

PERSON with PERSON-ID 101 is team member on PROJECT with PROJECT-ID P1
PERSON with PERSON-ID 101 is team member on PROJECT with PROJECT-ID P2 ***no***
PERSON with PERSON-ID 221 is team member on PROJECT with PROJECT-ID P1
PERSON with PERSON-ID 223 is team member on PROJECT with PROJECT-ID null ***no***
PERSON with PERSON-ID null is team member on PROJECT with PROJECT-ID P3 ***no***

The second and last two examples that were valid for UOD 1 are not valid for UOD 2. The constraints for the UOD 2 example set are:

A PERSON can work on only one PROJECT.
A PERSON must be a team member on a PROJECT.
A PROJECT may include many PEOPLE as team members.
A PROJECT must include at least one PERSON as a team member.

These types of conditions are very common when models of UODs throughout an enterprise are managed in an integrated manner. This leaves us with basically three options to resolve this condition: 1. create a new elementary fact type for UOD 2 with associated constraints, 2. modify the business rules to be consistent, 3. manage the same elementary fact and the differing constraints for each UOD.

In option 1; create a new elementary fact, the following fact types may be created:

PERSON is a team member on a UOD 1 PROJECT.
PERSON is a team member on a UOD 2 PROJECT.

This option is not desireable because it creates artificial facts in the enterprise which add little or no additional knowldge about the enterprise. It is preferred to manage a single elementary fact about the enterprise, i.e A Person is a team member on Project. This simple fact sufficiently expresses the desired knowledge about the enterprise. Also, one could argue that if a new fact (as shown in the example) were created, it is just another representation of the original elementary fact.

Option 2; modify the business rules to be consistent, should be evaluated when conditions like this arise. Remember, one of the fundamental reasons for managing information models at the enterprise level is to acheive a conceptually integrated enterprise.The question to be answered at this point is - Are these different rules simply an anomoly or is there a valid reason for them? If they are an anomoly, an opportunity exists to modify them. If they are valid differences, the rules for that fact type must be managed in the context of the UOD in which it participates.

Option 3 is the preferred option because inevitably the example condition will exist and the constraints of a fact type will vary by UOD. This approach forces the consideration of rules applied to each fact as it is applied to UODs throughout the enterprise. It quickly identifies coneptually equivalent information which is used differently within the enterprise.

5.1 The Fact Reuse Process

Steps involved in fact reuse are as follows:

1. Identify existing objects to be included in the new model.

2. Retrieve from the information knowledge base all existing facts in which these objects participate.

3. Select those facts to be included in the new model, and abstract them to form the basis of the new model.

For example, suppose the following facts are registered for the object named PART:

FT001	PART has a descriptive name of LONG PART NAME
FT002	PART was designed by an ENGINEER
FT003	PART was designed at DESIGN LABORATORY
FT004	PART is described by DRAWING
FT005	PART is manufactured at PRODUCTION AGENCY
FT006	PART was approved on a DATE
FT007	PART has waste stream described by WASTE STREAM DESCRIPTION
FT008	PART has a WEIGHT
FT009	PART has length of LINEAR DIMENSION

Suppose a new model is to be created for an application that characterizes part material to determine potential disposal hazards. The modeler might select facts FT001, FT002, FT004, and FT007. All registered knowledge about each fact and related objects is extracted from the registered fact base to form the basis of the new model. This knowledge includes the proper object references (identifiers), examples, constraints (as used by UODs), reference material describing the origin of each fact, and the owners of the fact descriptions. The registered information is guaranteed to be complete and accurate. The more facts which can be included from the fact base, the smaller the job of creating

the new model will be. A high quality model can be produced in less time, with the certainty of integration with existing models.

6.0 Fact Management Tools

The process of managing models in an enterprise can be made more efficient with automated tools. Automated utilities to assist in fact registration and reuse are needed to realistically implement the concepts presented in this paper. Configuration management of the models is also desired. Two new tools are now being prototyped at Sandia National Laboratories. The "Fact Manager" is a tool to provide object and fact registration, and constraint enforcement,across the knowledge base. Fact Manager is intended for use by data administrators to assist them in model management and fact registration processes.

Another tool under development is the "Fact Designer". Fact Designer is intended to support modelers and database designers as they go through the processes of CSDP and fact reuse. Its key functions are the ability to browse registered objects and facts, select objects and facts to be included a new model, and produce standard information model reports. In the future it will be able to import and export enterprise information to and from other automated design tools such as Bachman or Information Engineering Facility (IEF).

7.0 Events, Triggers, Processes

NIAM is currently being extended to include event, trigger, and process definition. While still in the formative stages, this approach is very exciting, and is a natural extension of the NIAM CSDP. Events, triggers, and processes can be defined in terms of their effect upon NIAM facts. Real-world events can be traced to their effects on specific facts, and vice versa. Triggers can be matched to required processes in implemented applications. Once this is done, a complete "Enterprise Model" emerges, which can be used to support the day-to-day business of the enterprise, and to run wide-ranging "what-if" analyses based on changes to fact parameters or processes. The enterprise model is capable of supporting any existing implementation paradigm, including object-oriented paradigms.

Sandia National Laboratories is proceeding in FY94 to extend its NIAM-based capabilities to include events, triggers, and processes and validate the process through its application in a number of diverse subject areas. This effort will build upon the fact management tools and administrative techniques described above.

8.0 Summary

Effective Enterprise Information Management is an elusive goal, but one that will be at the center of enterprise integration efforts in the future. It is the key to unifying "islands of information", providing advanced reporting and decision support services, making the development of new systems cost-effective, and coping with the information contained in "legacy" systems. Many attempts have been made to accomplish EIM, most of them resulting in high costs, schedules which are not met, hard-to-maintain documentation, and management dissatisfaction. The automated tools used in these efforts work best with single applications, and have failed when used for enterprise integration. Management is caught between these failures and the clear need to continue the effort. A new approach is needed. The natural language based methodolgy (NIAM) can provide a solid

foundation for enterprise modeling but areas involving the management of multiple Universes of Discourse need to be better defined. A formal prescription for defining events, triggers, and processes is also needed.

The EIM approach of the future must accomplish integration without sacrificing the bottom line. It must be built into the process of application development, yet remain neutral across applications and automated tools. It must be understandable to all users in the enterprise, including managers, strategic planners, system operators, end users, and information modelers. The NIAM method, when coupled with the techniques of enterprise-wide data administration, collection of metrics, automated tools that produce application code, and automated support tools, can fulfill these objectives. EIM can be based upon a commonly understood language shared by all of its "Universes Of Discourse" - structured natural language.

References and Bibliography

1. Goodhue, Dale L.,Kirsch, Laurie J., Quillard, Judith A., Wybo Michael D. "Strategic Data Planning: Lessons From the Field," MIS Quarterly, March 1992, pp. 11-34.
2. Moriarity, Terry "Business Rule Analysis," Database Programming and Design, April 1993, pp.66-69.
3. Nijssen, G.M., Twine, Steven "An Effective Design Method for Relational Databases," Advanced NIAM Course Material, 1990.
4. Nijssen, G.M., Twine, Steven "A Framework for Information System Development Methodologies," Advanced NIAM Course Material, 1990.
5. Nijssen, G. M., Halpin, T. A. "Conceptual Schema and Relational Database Design: A Fact Oriented Approach," Prentice Hall of Australia Pty Ltd., 1989.
6. Nijssen, G. M., Yunker K. W. "Information Model Design for a NIAM Information Model Management Tool. Prepared for Sandia National Laboratories," Nijssen Adviesbureau voorInformatica, 1992.
7. Texas Instruments Inc. "Information Strategy Planning," IEF Student Guide, Texas Instruments Release 4.0-1.0.
8. Wintraecken, J.J.V.R. "The NIAM Information Analysis Method: Theory and Practice," Kluwer Academic Publishers, PO Box 17, 3300 AA Dordrecht, The Netherlands, 1990.

DATE
FILMED
12/20/93
END