Appeared in: *Proceedings of the IEEE Conference on Systems, Man and Cybernetics,* 1993.

Issues in Enterprise Modelling

Mark S. Fox

Department of Industrial Engineering, University of Toronto
4 Taddle Creek Road, Toronto, Ontario M5S 1A4 CANADA
tel: +1-416-978-6823; fax: +1-416-978-3453; internet: msf@ie.utoronto.ca

Abstract: Computerization of enterprises continues unabated and so does the cost of software. The availability of a generic, common-sense enterprise model is necessary if we are to reign in costs. But in order to construct useful Generic Enterprise Models (GEM) there are a number of issues that have to be addressed. In this paper we explore the following issues: Is there such a thing as a generic enterprise model? Can the terminology be precisely defined? Does all knowledge need to be explicit? Need there be a single, shared enterprise model? How can we determine which is a better enterprise model? Can an enterprise model be consistent? Can an enterprise model be created and kept current? Will the organization accept an enterprise-wide model? We then briefly describe the TOVE project, which attempts to address many of these issues.

1.0 Introduction

As described in a recent report on Agile Manufacturing [Nagel & Dove 92], if an industrial organization is to compete in the coming decade, they must produce products that are: of consistently high quality throughout the product's life, customised to local market needs, open in that they may be integrated with other products, environmentally benign, and technically advanced. The key to achieving these capabilities is "agility". Agility implies the ability to: continuously monitor market demand, quickly respond by providing new products, services and information, quickly introduce new technologies, and quickly modify business methods. But achieving agility requires far greater *integration* of functions within the enterprise, and between enterprises, than has ever been achieved; enterprises must be task oriented as opposed to organisation oriented; expertise must flow freely across the enterprise to where it is needed.

Integration is a step along the road to agility. Yet it contradicts decades of management science teachings. We have been taught that in order to cope with the complexity of enterprises, we have to decompose them into manageable pieces; each piece having minimal interaction with the others. But the decomposition impedes the free flow of information and knowledge, and the coordination of actions. In order to breakdown these organizational barriers, Hansen [91] has identified five principles of integration:

1. "When people understand the vision, or larger task, of an enterprise and are given the right information, the resources, and the responsibility, they will 'do the right thing'."

2. "Empowered people - and with good leadership, empowered groups - will have not only the ability but also the desire to participate in the decision process."

3. "The existence of a comprehensive and effective communications network ... This network must distribute knowledge and information widely, embracing the openness and trust that allow the individual to feel empowered to affect the 'real' problems."

4. "The democratization and dissemination of information throughout the network in all directions irrespective of organizational position ... ensures that the Integrated Enterprise is truly integrated."

5. "Information freely shared with empowered people who are motivated to make decisions will naturally distribute the decision-making process throughout the entire organization."

These principles focus on two major issues: 1) how to motivate employees, and 2) how to provide employees with the right information to do their job. But in achieving the latter, there is a limit to how many meetings you can attend, memos you can read, and trips you can make! The question then is how can technology aid integration?

Over the last 10 years there has been a shift in how we view the operations of an enterprise. Rather than view the enterprise as being hierarchical in both structure and control, a distributed view where enterprise units communicate and cooperate in both problem solving and action has evolved [Fox 81]. To achieve integration it is necessary that units of

the enterprise, be they human or machine based, be able to understand each other. Therefore the requirement exists for a representation in which enterprise knowledge can be expressed. Minimally the representation provides a language for communicating among units, such as design, manufacturing, marketing, field service, etc. Maximally the representation provides a means for storing knowledge and employing it within the enterprise, such as in computer-aided design, production control, etc.

The problem that we face today, is that the legacy systems to support enterprise functions were independently created, consequently they do not share the same representations. This has led to different representations of the same enterprise knowledge and the inability of these functions to share knowledge. We call this the *Correspondence Problem*: What is the relationship among concepts that denote the same thing but have different names? It is common for enterprises, especially those that are geographically dispersed to use different names to refer to the same concept. No matter how rationale the idea of renaming them is, organisational barriers impede it.

Secondly, these representations lack an adequate specification of what the terminology means (aka semantics). This leads to inconsistent interpretations and uses of the knowledge. Lastly, the cost of designing, building and maintaining a data model of enterprise knowledge is large. Each tends to be unique to the enterprise; terminology is enterprise specific.

As a solution to this problem, there has been an increasing interest in Generic Enterprise Models (GEM). A GEM is a data dictionary that defines the classes of entities (or objects) that are generic across a type of enterprise, such as manufacturing, and can be employed (aka instantiation) in defining a specific enterprise. It is believed that if one starts with a GEM, the time and cost of producing an instantiation of the model will be reduced significantly. Though much work has gone into the creation of GEMs, few have reflected upon the issues that arise in their creation and use. In the following, we explore a number of issues surrounding the creation of a Generic Enterprise Model.

2.0 Enterprise Modelling Issues

2.1 Is there such a thing as a generic enterprise model?

Yes! There exists significant amounts of knowledge that is generic across many applications. The identification and formalization of generic knowledge has come to be called "Ontological Engineering" [Gruber 93]. An ontology is a formal description of entities and their properties, relationships, constraints, behaviours. Entities are classified into one or more taxonomies.

In trying to construct an ontology that spans enterprise knowledge, the first question is where to start. Brachman provides a stratification of representations [Brachman 79]:

Implementation: physical representation of data

Logical: logical interpretation of the physical representation.

Conceptual (aka Epistemological): primitives for representing the components of a concept: properties, structure, relations, generalization, association.

Generic: domain independent concepts such as time, causality, action, space, etc.

Application (aka Lexical): primitives are application dependent and may change meaning as knowledge grows.

The following diagram depicts the last three levels with examples of the type of knowledge that is represented at each. Note that the application level is re-labeled the enterprise level. Secondly, the division between levels is somewhat artificial in that each level may be further stratified. Determining what concepts should be in the generic level versus the enterprise level is based on their generality.

The *conceptual level* provides the building blocks for defining concepts. The basic unit of representation is an *object* for which is defined:

- **Properties:** Cardinality, Type
- **Relationships:** Range restrictions
- **Generalization/Specialization** hierarchies
- **Classification:** Prototypical descriptions vs. instances

The conceptual level received much attention in the 1970s, with the development of knowledge representation languages such as FRL [Roberts & Goldstein 77], KLONE [Brachman 77], KRL [Bobrow & Winograd 77], NETL [Fahlman 77], and SRL [Fox 79],. Many of the concepts investigated in these system have formed the basis of semantic data modelling in databases. More recently, the conceptual level has been formalized as what is now called "terminological logic" [Brachman & Schmolze 85].

The *generic level* provides ontologies for concepts common across many domains. Generic level representations include concepts such as:

- Time [Allen 83],
- Causality [Rieger & Grinberg 77], [Bobrow 85],

- Activity [Sathi et al. 85], and
- Constraints [Fox 83] [Davis 87].

Consider the representation of time. Time is represented by points, periods and relations. A time-point lies within an interval {<tmin, tmax> | tmin ≤ tmax, tmin, tmax Œ N}. A time-period is bounded by a start and end time-point {<TP1, TP2> | tmin1 ≤ tmax2, TP1, TP2 Œ TP}. We use Allen's [83] temporal relations to describe the relationships between time-points and/or time-periods. We present the thirteen possible temporal relationships and refer to Allen's paper for the transitivity table of these temporal relations.

One of the largest efforts underway to create an integrated set of generic representations is the CYC project at MCC [Lenat & Guha 90]. At the University of Toronto, the TOVE project has focused on the creation of ontologies for industrial applications, covering activities, states, time, causality, resources, quality, cost, organization structure, etc. [Fox 92].

The *enterprise level* provides a data dictionary of concepts (aka reference model) that are common across various enterprises, such a products, materials, personnel, orders, departments, etc. At the enterprise level, various efforts exist in standardizing representations. For example, since the 1960's IBM's COPIC's Manufacturing Resource Planning (MRP) system has had a shared enterprise model. In fact, any MRP product contains an enterprise model. Recently, several efforts have been underway to create more comprehensive enterprise model, including:

CAMI: A US-based non-profit group of industrial organizations for creating manufacturing software and modelling standards.

ICAM: A project run by the Materials Lab. of the US Air Force [Martin & Smith 83] [Smith et al. 83].

IWI: A reference model developed at the Institut fur Wirtschaftsinformatik, Universitat des Saarlandes, Germany [Scheer 89].

The following are the basic relations and objects in their range defined for the "part" concept in the ICAM model from the *design* perspective [Martin & Smith 83]:

- **IS CHANGED BY:** Part Change (105) (also shown as "is modified by")
- **APPEARS AS:** Next Assembly usage item (119) (also shown as "is referenced as").
- **HAS:** Replacement part (143).
- **HAS SUBTYPE (IS):** Parts list item (118), Replacement part (143).
- **IS USED AS:** Next Assembly Usage (40), Advance material notice item part (144), Configuration list item (170).
- **IS TOTALLY DEFINED BY:** Drawing (1).
- **IS LISTED BY (LISTS):** Configuration list (84).
- **IS USED IN:** Effectivity (125).
- **IS FRABRICATED FROM:** Authorized material (145).

The following ar e the basic relations and objects they are linked to for a "part" from a *manufacturing* perspective:

- **HAS:** N.C. Program (318), Material issue (89), Component part (299), Alternative part (301), Part/process specification use (255), Material receipt (87), Work package (380), Part tool requirement (340), Part requirement for material (397), Standard routing use (254), Image part (300), Part drawing (181).
- **IS ASSIGNED TO (HAS ASSIGNED TO IT):** Index (351).
- **IS DEFINED BY (DEFINES):** Released engineering drawing (12).
- **IS SUBJECT OF:** Quote request (90), Supplier quote (91).
- **IS TRANSPORTED BY:** Approved part carrier (180).
- **IS RECEIVED AS:** Supplier del lot (309).
- **APPEARS AS:** Part lot (93), Ordered part (188), Serialized part instance (147), Scheduled part (409), Requested purchase part (175).
- **CONFORMS TO:** Part specification (120).
- **IS INVERSE:** Component part (299), Alternate part (301), Section (363), End item (5), Configured item (367), Image part (300).
- **IS USED AS:** Component part callout (230), Process plan material callout (74).
- **IS SUPPLIED BY:** Approved part source (177).
- **MANUFACTURE IS DESCRIBED BY:** Process plan (415).
- **SATIFIES:** End item requirement for part (227).
- **IS REQUESTED BY:** Manufacturing request (88).
- **IS STORED AT:** Stock location use for part (227).
- **IS SPECIFIED BY:** BOM Item (68).

This is only the tip of the iceberg. If one were to develop a complete GEM at the enterprise level, its sheer size would overwhelm the abilities of any database manager or knowledge engineer. There is a point at which further elaboration tends to obfuscate rather than enhance the model. On the other hand, if there is not enough detail, then its value may be limited. We will revisit this issue in section 2.7.

2.2 Can the terminology be precisely defined?

> "It is certainly praiseworthy to try to make clear to oneself as far as possible the sense one associates with a word. But here we must not forget that not everything can be defined" Gottlob Frege.

Prior to the advent of GEMs, an application's data model was defined in a database system's data dictionary. The creation of the data dictionary was and continues to be the responsibility of the database administrator who works with the end users. In the worst case, definitions for each of the objects, attributes and relations in the dictionary are not available, and their interpretation can only be derived by looking at how an application used the information. Better managed data dictionaries include definitions, usually written in a natural language such as english. Due to the inherent ambiguities of natural language, even these definitions may be interpreted differently by each user. If we are to create truly sharable GEMs, we need the ability to precisely state the meaning of each object, attribute and relation.

Precise definitions can be constructed. Through the use of logic, we can define more precisely the meaning of each object, attribute and relation as needed. Definitions may be hierarchical and circular. Hierarchical in the sense that enterprise level concept are defined in terms of generic level concepts. Circular in that enterprise level concepts are defined in terms of other concepts at the same level, and vice versa! Many, if not most, definitions can be represented using first order logic. Some definitions may require high order languages, but it is probably the case most things can exist in a first order language.

Consider the temporal relations introduced in the previous section. The following are definitions of two variations of the *before* relation:

TimePoint1 is *possibly before* TimePoint2 IF tmin1 < tmax2 (EQ 1)

TimePoint1 is *strictly before* TimePoint2 IF tmax1 < tmin2 (EQ 2)

Tmin1 and tmax1 bound the interval in which time point 1 is located. The first axiom states that for TimePoint1 to be possibly before TimePoint2, there must exist at least one point in time in TimePoint1's associated interval that is less than some point in time in TimePoint2's associated interval. This is true iff tmin1 < tmax2.

2.3 Does all knowledge need to be explicit?

The usefulness of an instantiated GEM is determined by the queries it can answer. Consider a model with an SQL interface. Knowledge is explicitly represented if it can be retrieved using a simple SELECT. That is, the knowledge is represented explicitly and only needs to be retrieved. Knowledge is represented implicitly if it requires a more complex query to retrieve it. For example, it may require one or more JOINs combined with SELECTs. This is equivalent to performing deduction. For example, if the model contains a 'works-for' relation and it is explicitly represented that Joe 'works-for' Fred, and that Fred 'works-for' John, then the obvious deduction that Joe 'works-for' John (indirectly) is not represented explicitly in the model but must be deduced.

We distinguish between a model that includes axioms that support deduction, versus a model without axioms where deductions are specified by the query. In the former case, the model would be able to deduce that Joe works-for John in response to a query asking who does Joe work for. In the latter case, the user would have to specify a complex query which would include as many joins as necessary to travel along the works-for relation. Since the user does not know at the outset the depth of the works-for path, they may not get the information they were looking for. We call a model which includes axioms an Axiomatised Enterprise Model (AEM). An AEM that includes a deduction engine (i.e., theorem prover) has been called either a knowledge base or a deductive database. We will refer to it as a Deductive Enterprise Model (DEM). The lack of a deductive capability forces users to spend significant resources on programming each new report or function that is required.

So far we have discussed the deductive capability of a model without reference to the nature of the axioms or rules used in performing the deductions. We say a DEM possesses Common-Sense (DEM_{CS}) if its axioms define the meaning of the terms in the ontology. By Common-Sense, we mean that the axioms enable the model to deduce answers to questions that one would normally assume can be answered if one has a "commons-sense" understanding of the enterprise.

In summary, the design, creation and maintenance of software is fast becoming the dominant cost of automation. A significant portion of these costs is for software that provides answers deduced from the contents of the enterprise model. Many of these questions could be answered automatically if the enterprise model had the "common sense" to answer them!

2.4 Need there be a single, shared enterprise model?

Not all knowledge has to be represented generically, only that which is shared among units of the enterprise, and that too may be specialized. Units of an enterprise evolve representations and procedures that are tailored to their roles and goals. The tailoring is usually necessary to achieve higher degrees of productivity and quality. Consequently, formalized models maximally affect what is communicated among enterprise units, and minimally affect how information/knowledge is represented within units.

Even the interchanges among units in the enterprise neither require nor desire a single integrated model as a basis of communication. As shown in the figure above, there may be one language using for communication between engineering and manufacturing, and a different one for engineering and marketing. But all units will share some core language. Though the artificiality of the enterprise implies the possibility of an integrated model, reality tends to differ. Integrated models are really a lattice of models that are specialized to the needs of subsets of enterprise units.

2.5 How can we determine which is a better enterprise model?

Given the many efforts seeking to create a GEM, there has never been a well defined set of criteria with which these efforts could be evaluated! In fact, there is no objective means by which one can compare one GEM with another. Following are what we believe should be the characteristics of a representation:

Generality: To what degree is the representation shared between diverse activities such as design and troubleshooting, or even design and marketing?

Competence: How well does it support problem solving? That is, what questions can the representation answer or what tasks can it support?

Efficiency: Space and inference. Does the representation support efficient reasoning, or does it require some type of transformation?

Perspicuity: Is the representation easily understood by the users? Does the representation "document itself?"

Transformability: Can the representation be easily transformed into another more appropriate for a particular decision problem?

Extensibility: Is there a core set of ontological primitives that are partitionable or do they overlap in denotation? Can the representation be extended to encompass new concepts?

Granularity: Does the representation support reasoning at various levels of abstraction and detail?

Scalability: Does the representation scale to support large applications?

These criteria bring to light a number of important issues and risks. For any set of functions, how can we determine if the integrating model is *functionally complete*? A model is functionally complete if it contains the types of information necessary for a function to perform its task. Are functionally complete models specifiable? One way of specifying a model's functional requirements is as a set of questions that the model must be able to answer. We call this the *competency* of a model.

Another problem is where the representation ends and inference begins? Consider the competence criterion. The obvious way to demonstrate competence is to define a set of questions that can be answered by the representation. If no inference capability is to be assumed, then question answering is strictly reducible to "looking up" an answer that is represented explicitly. In contrast, Artificial Intelligence representations have assumed at least inheritance as a deduction mechanism. In defining a shared representation, a key question then becomes: should we be restricted to just an terminology? Should the terminology assume an inheritance mechanism at the conceptual level, or some type of theorem proving capability as provided, say, in a logic programming language with axioms restricted to Horne clauses (i.e., Prolog)? What is the *deductive capability* that is to be assumed by a reusable representation?

The efficiency criterion is also problematic. Experience has demonstrated that there is more than one way to represent the same knowledge, and each representation does not have the same complexity when answering a specific class of questions. Consequently, we cannot assume that a representation will partition the space of concepts, but there will exist overlapping representations that are more efficient in answering certain questions. Furthermore, the deductive capability provided with the representation affects the store vs. compute trade-off. If the deduction mechanisms are taken advantage of, certain concepts can be computed on demand rather than stored explicitly.

The ability to validate a proposed representation is critical to this effort. The question is: how are the criteria described above operationalised? The *competence* of a representation is concerned with the span of questions that it can answer. We propose that for each category of knowledge, a set of ques-

tions be defined that the representation can answer. Given a conceptual level representation and an accompanying theorem prover (perhaps Prolog), questions can be posed in the form of queries to be answered by the theorem prover. Given that a theorem prover is the deduction mechanism used to answer questions, the *efficiency* of a representation can be defined by the number of LIPS (Logical Inferences Per Second) required to answer a query. Validating *generality* is more problematic. This can be determined only by a representation's consistent use in a variety of applications. Obviously, at the generic level we strive for wide use across many distinct applications, whereas at the application level, we are striving for wide use within an application.

2.6 Can an enterprise model be consistent?

The assumption that enterprise knowledge can be globally consistent is ridiculous. By definition, an information system based on a distributed architecture will abound in inconsistent information. Tailoring and local context leads to ambiguities and inconsistencies in the content of what is stored and communicated. How to manage inconsistency so that it does not adversely affect operations is the problem that has to be solved.

One way of approaching this is to identify subsets of knowledge that must remain consistent among a set of "consenting" agents in the information network. Changes to this knowledge must be managed so that inconsistencies do not arise.

2.7 Can an enterprise model be created and kept current?

Enterprises are dynamic and undergo continuous change. Consequently, a process for managing the evolution of the model is required. Since the competence of a model is specified by the activities that use it, it follows that model management is an activity-based process. The information requirements of activities determine data spheres and their contents. A data sphere is a set of information that is shared by functionally-related agents. Groupings of activities lead to data spheres whose model is a point in the model lattice.

Since enterprise activities are the result of enterprise design, model specification is the outcome of enterprise design. Without an adequate process - and possibly a theory - of enterprise design, the construction of an integrated model will be either expensive or impossible. Emerging methods for enterprise analysis and possibly design include:

- GRAI: Universite de Bordeaux.
- CIM-OSA: A reference model being developed by the ACIME group of ESPRIT in Europe [Esprit 90].
- PERA: Purdue Enterprise Reference Architecture [Williams 91].

2.8 Will the organization accept an enterprise-wide model?

There is a belief that an integrated model cannot be superimposed upon an enterprise. Enterprises are both artificial and natural. Artificial in that formal structures and systems exist within the enterprise by design. Natural in that systems evolve in response to the inadequacies of the design due to changing market conditions, technologies, knowledge, etc. The artificiality of an enterprise admits the specification and utilization of an integrated model. Its adoption is imposed by the enterprise's formal structures.

3.0 Conclusion

Computerization of enterprises continues unabated. The amount of software is increasing while its cost is not decreasing. The availability of a generic, common-sense enterprise model is necessary if we are to reign in costs. But in order to construct useful Generic Enterprise Models there are a number of issues that have to be addressed. Foremost is the transition of the efforts from poorly principled data modelling into principled engineering.

4.0 Acknowledgments

This research is supported in part by an NSERC Industrial Research Chair in Enterprise Integration, Carnegie Group Inc., Digital Equipment Corp., Micro Electronics and Computer Research Corp., Quintus Corp., and Spar Aerospace Ltd.

5.0 References

[Allen 83] Allen, J.F. Maintaining Knowledge about Temporal Intervals. *Communications of the ACM.* 26(11):832-843, 1983.

[Bobrow 85] Bobrow, D.G. *Qualitative Reasoning About Physical Systems.* MIT Press, 1985.

[Bobrow & Winograd 77] Bobrow, D., and Winograd, T. KRL: Knowledge Representation Language. *Cognitive Science.* 1(1), 1977.

[Brachman 77] Brachman, R.J. *A Structural Paradigm for Representing Knowledge.* PhD thesis, Harvard University, 1977.

[Brachman & Schmolze 85] Brachman, R.J., and Schmolze, J.G. An Overview of the KL-ONE Knowledge Representation Systems. *Cognitive Science.* 9(2), 1985.

[Davis 87] Davis, E. Constraint Propagation with Interval Labels. *Artificial Intelligence.* 3281-331, 1987.

[Esprit 90] ESPRIT-AMICE. CIM-OSA - A Vendor Independent CIM Architecture. *Proceedings of CINCOM 90*, pages 177-196. National Institute for Standards and Technology, 1990.

[Falhman 77] Fahlman, S.E. *A System for Representing and Using Real-World Knowledge.* PhD thesis, Massachusetts Institute of Technology, 1977.

[Fox 79] Fox, M.S. On Inheritance in Knowledge Representation. *Proceedings of the International Joint Conference on Artificial Intelligence.* 95 First St., Los Altos, CA 94022, 1979.

[Fox 81] Fox, M.S. An Organizational View of Distributed Systems. *IEEE Transactions on Systems, Man, and Cybernetics.* SMC-11(1):70-80, 1981.

[Fox 83] Fox, M.S. *Constraint-Directed Search: A Case Study of Job-Shop Scheduling.* PhD thesis, Carnegie Mellon University, 1983. CMU-RI-TR-85-7, Intelligent Systems Laboratory, The Robotics Institute, Pittsburgh.

[Fox 92] Fox, M.S., (1992), "The TOVE Project: Towards a Common Sense Model of the Enterprise", in Enterprise Integration, C. Petrie (Ed.), Cambridge MA: MIT Press.

[Fox et al. 89] Fox, M.S., Reddy, Y.V., Husain, N., McRoberts, M. Knowledge Based Simulation: An Artificial Intelligence Approach to System Modeling and Automating the Simulation Life Cycle. *Artificial Intelligence, Simulation and Modeling.* In Widman, L.E., John Wiley & Sons, 1989.

[Gruber 93] Gruber, T.R. Toward Principles for the Design of Ontologies Used for Knowledge Sharing. Technical Report, Knowledge Systems Laboratory, Stanford University, 1993.

[Hansen 91] Hansen, W.C. The Integrated Enterprise. In *Foundations of World-Class Manufacturing Systems: Symposium Papers.* National Academy of Engineering, 2101 Constitution Ave, N.W., Washington DC, 1991.

[Lenat & Guha 90] Lenat, D., and Guha, R.V. *Building Large Knowledge Based Systems: Representation and Inference in the CYC Project.* Addison Wesley Pub. Co., 1990.

[Martin & Smith 83] Martin, C., and Smith, S. *Integrated Computer-aided Manufacturing (ICAM) Architecture Part III/Volume IV: Composite Information Model of "Design Product" (DES1).* Technical Report AFWAL-TR-82-4063 Volume IV, Materials Laboratory, Air Force Wright Aeronautical Laboratories, Air Force Systems Command, Wright-Patterson Air Force Base, Ohio 45433, 1983.

[Nagel & Dove 92] Nagel, R., and Dove, R., "Agile Manufacturing", Lehigh University, 1992.

[Rieger & Grinberg 77] Rieger, C., and Grinberg, M. *The Causal Representation and Simulation of Physical Mechanisms.* Technical Report TR-495, Dept. of Computer Science, University of Maryland, 1977.

[Roberts & Goldstein 77] Roberts, R.B., and Goldstein, I.P. *The FRL Manual.* Technical Report MIT AI Lab Memo 409, Massachusetts Institute of Technology, 1977.

[Sathi et al. 85] Sathi, A., Fox, M.S., and Greenberg, M. Representation of Activity Knowledge for Project Management. *IEEE Transactions on Pattern Analysis and Machine Intelligence.* PAMI-7(5):531-552, September, 1985.

[Scheer 89] Scheer, A-W. *Enterprise-Wide Data Modelling: Information Systems in Industry.* Springer-Verlag, 1989.

[Smith et al. 83] Smith, S., Ruegsegger, T., and St. John, W. *Integrated Computer-aided Manufacturing (ICAM) Architecture Part III/Volume V: Composite Function Model of "Manufacture Product" (MFG0).* Technical Report AFWAL-TR-82-4063 Volume V, Materials Laboratory, Air Force Wright Aeronautical Laboratories, Air Force Systems Command, Wright-Patterson Air Force Base, Ohio 45433, 1983.

[Williams 91] Williams, T.J., and the Members, Industry-Purdue University Consortium for CIM. *The PURDUE Enterprise Reference Architecture.* Technical Report Number 154, Purdue Laboratory for Applied Industrial Control, Prudue University, West Lafayette, IN 47907, 1991.