A COMMON-SENSE MODEL OF THE ENTERPRISE

Mark S. Fox, John F. Chionglo, Fadi G. Fadel

Department of Industrial Engineering,
University of Toronto
4 Taddle Creek Road, Toronto, Ontario M5S 1A4 Canada
tel: +1-416-978-6823, fax: +1-416-971-1373, internet: msf@ie.utoronto.ca

ABSTRACT

There is a paradigm shift towards a distributed and integrated enterprise. Currently, computer systems that support enterprise functions were created independently. This hampers enterprise integration Therefore, there is a need for a computer based data model which provides a shared and well defined terminology of an enterprise, and has the capability to deductively answer common sense questions.

This paper discusses how TOVE tackles these needs by defining a framework for modeling generic level representations such as activities, time, and resources. Since there has never been a well-defined set of criteria to evaluate such models, this paper also introduces a set of evaluation criteria which may be used to evaluate modelling efforts.

INTRODUCTION

Over the last 10 years there has been a shift in how we view the operations of an enterprise. Rather than view it as being hierarchical in both structure and control, a distributed view where organizational units communicate and cooperate in both problem solving and action has evolved [7] [8]. Enterprise Integration is concerned with how to improve the performance of distributed organizations and markets. It focuses on the communication of information and the coordination and optimization of enterprise decisions and processes in order to achieve higher levels of productivity, flexibility and quality. 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 language in which enterprise knowledge can be expressed. Minimally the language provides a means of communicating among units, such as design, manufacturing, marketing, and field service. Maximally the language provides a means of representing knowledge for use in the enterprise, such as in computer-aided design, and production control.

Currently, computer systems that support enterprise functions were created independently. This leads to three problems. Firstly, the functions do not share the same representations (i.e. different representations of the same enterprise knowledge); hence, they are unable to share knowledge. Secondly, the representations were defined without an adequate specification of what the terminology means (a.k.a. semantics); hence, the interpretations and uses of the knowledge are inconsistent. Thirdly, the representations are passive. They do not have the capability to automatically deduce the obvious about what it is representing. For example, if the representation 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) cannot be made within the representation system. The lack of a `common-sense' deductive capability forces users to spend significant resources on programming each new report or function that is required.

The goal of the TOVE (TOronto Virtual Enterprise) project is to create a data model that: 1) provides a shared terminology for the enterprise that each agent can jointly understand and use, 2) defines the meaning of each term (a.k.a. semantics) in a precise and as unambiguous manner as possible, 3) implements the semantics in a set of axioms that will enable TOVE to automatically deduce the answer to many “common sense” questions about the enterprise, and 4) defines a symbology for depicting a term or a concept in a graphical context

In the following sections, we introduce a set of evaluation criteria by which representations may be measured. Finally, we describe our efforts to date in creating a more formal and computable terminology and semantics of activities and states, time, and resources.

EVALUATION CRITERIA

Though there are a number of efforts underway to create enterprise models, it is not clear how to evaluate these efforts. Consider the ICAM [10] reference model. Like the IWI [13] model, it is an application level data model whose entities, properties and relationships are specific to manufacturing. But there are problems. First, the meanings of the relations are ambiguous and at best provided in a descriptive form. Second, there is no way to know whether this is the "right" way of representing this information. Third, this is only the tip of the iceberg, if one were to develop a complete model for an enterprise, its sheer size would be beyond the abilities of any database manager or knowledge engineer to understand and use effectively. All of these models fall short of our goals.

Though all of these efforts seek to create a sharable representation of enterprise knowledge, there has never been a well defined set of criteria that these efforts should satisfy. 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 can be partitioned 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 example, where does the representation end and inference begin? 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 ability to validate a proposed representation is critical to this effort. The question is: how are the criteria described above operationalized? 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 questions 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.

THE TOVE PROJECT

We approach the first goal by defining a generic level representation. Generic concepts include representations of Time [2], Causality [11] [3], Activity [12], and Constraints [8] [5]. The generic level is, in turn, defined in terms of a conceptual level based on the 'terminological logic' of KLONE [4]. Application level representations will be defined in terms of the generic level.

We approach the second and third goals by defining a set of axioms (a.k.a. rules) that define common-sense meanings for the terminology. By common sense, we mean that the more obvious definitions/deductions about the entities and attributes in our ontology. What is an obvious deduction should be determined by a subset of questions used to determine the competence of a representation. Since there does not exist a standard for determining the competence of a model, we will define in English a set of questions and the axioms used to answer them.

TOVE is not only a research project but also a testbed. TOVE has been used to implement a *virtual company* whose purpose is to provide a testbed for research into enterprise integration. TOVE is implemented in C++ using the ROCK@+[TM] knowledge representation tool from Carnegie Group. TOVE operates "virtually" by means of knowledge-based simulation [9].

In the following sections, we describe the terminology and axioms, and outline the competence of the model.

Activity

Enterprises are action oriented, therefore the ability to represent action lies at the heart of all enterprise models. The CIM-OSA model stratifies action from the lowest level of a *function*, to an *enterprise activity* and up to a *business process*, the IWI representation defines function specific actions, and the Purdue PERA model has a two level representation composed of a *task* at a lower level and a *function* at the upper level. In TOVE, a single entity called an activity spans all of the above.

In TOVE, action is represented by the combination of an ***activity*** and its corresponding enabling and caused ***states***. An activity is the basic transformational action primitive with which processes and operations can be represented. An enabling state defines what has to be true of the world in order for the activity to be performed. A caused state defines what will be true of the world once the activity has been completed.

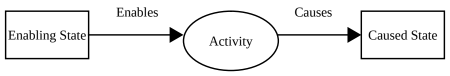


Figure 1. Activity-State Model

Status: The status of an activity is reflected in an attribute called status. We define the domain of an activity's status as a set of linguistic constants. ***dormant***: the activity is idle and has never been executing before. ***enabled***: the activity is executing. ***suspended***: the activity was executing and has been forced to an idle state. ***reEnabled***: the activity is executing again. ***completed***: the activity has finished.

State

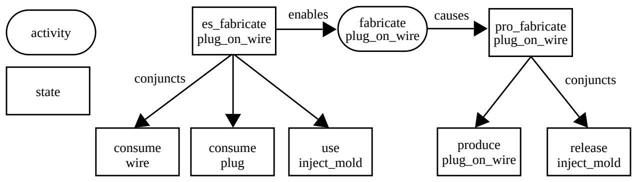


Figure 2. Activity-State Cluster

An activity along with its enabling and caused states is called an *activity cluster.* The state tree linked by an ***enables*** relation to an activity specifies what has to be true in order for the activity to be performed. The state tree linked to an activity by a ***causes*** relation defines what will be true of the world once the activity has been completed (see Figure 2.)

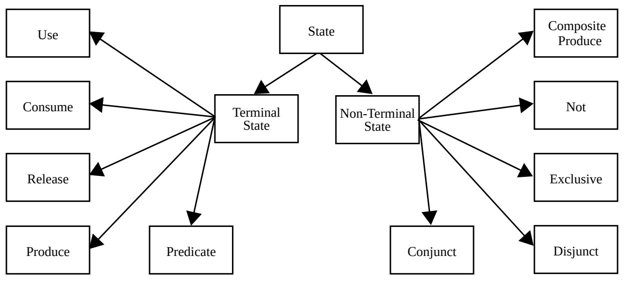


Figure 3. State Taxonomy

There are two types of states: ***terminal*** and ***non-terminal***.

Terminal States: ***Use*** signifies that a resource is to be used, but not consumed, by the activity, and will be released once the activity is completed***.*** ***Consume*** signifies that a resource is to be used/consumed by the activity and will not exist once the activity is completed. ***Release*** signifies that a resource, which has been designated as being used is now available for use/consumption elsewhere. ***Produce*** signifies that a resource, that did not exist prior to the performance of the activity, has been created by the activity. ***Predicate*** represents an arbitrary predicate that evaluates to either true or false.

Non-terminal states: (allows for the boolean combination of states). ***Conjunctive/Disjunctive*** specifies that all substates / at least one substate must be satisfied. ***Exclusive*** specifies that only one substate must be satisfied. ***Not*** specifies that the substate must not be satisfied. ***Composite Produce*** signifies that a resource, that did not exist prior to the performance of the activity, has been created by the activity. This resource includes other materials that are only being used for a limited time and will be released by another activity.

Except for the composite produce, each of these states can be further classified as being ***discrete*** or ***continuous***.

Status: The status of a state is reflected in an attribute called status. We define the domain of a state's status as a set of linguistic constants. For example, the domain for discrete_consumption is: ***possible/not_possible***: a unit of the resource that the state ***consumes*** is available/not available at the time required. ***committed***: a unit of the resource that the state ***consumes*** has been reserved for consumption. ***enabled***: a unit of the resource that the state ***consumes*** is being consumed. ***completed***: unit of the resource that the state ***consumes*** had been consumed and is no longer needed.

Activity Abstraction

Activity clusters may be aggregated to form multiple levels of abstraction. "An activity is elaborated to an aggregate activity (an activity network), which then has activities" [12]. These activities are subactivities of the aggregate activity (see Figure 4.). The enabling and caused states are omitted from this diagram but they do exist. Just as activities can be abstracted, states can be abstracted in a similar manner.

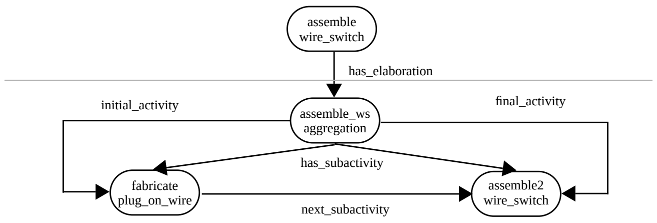
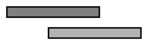
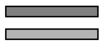
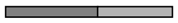
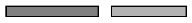
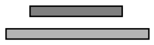
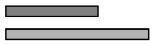
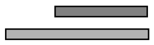


Figure 4. Activity Abstraction

Time

Time is represented by points, periods and relations. A time-point lies within an interval. A time-period is bounded by a start and end time-point. We use Allen's temporal relations [1] 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.

The temporal relation between a terminal state and an activity is specified by the user. The temporal relation between a non-terminal state and an activity may be automatically deduced using Allen's transitivity table for the thirteen temporal relations.

Relation	Time	Symbol	Inverse
X overlaps Y		o	oi
X same Y		=	=
X meets Y		m	mi
X meets Y		<	>
X during Y		d	di
X starts Y		d	di
X ends Y		d	di

<u>Manifestations</u>: A manifestation is a time-dependent description of an entity. Every instance of a state or activity may have one or more manifestations with an associated time-period over which the manifestation is true.

<u>Resources</u>

We view that "being a resource" is not innate property of an object, but is a property that is derived from the role an entity plays with respect to an activity. Accordingly, resources could be: *Machines* such as milling machines, computers etc.**,** *Electricity* consumed by an activity**,** *Materials* such as raw material, semi finished products etc., *Tools/Equipment* such as fixtures, cranes, chairs etc., *Capital* needed to perform an activity, *Human skill* when needed to perform an activity, and *Floor space* that is used by an activity. Following are some important properties of resources.

Divisibility**:** A resource is said to be temporal-divisible if the use of a resource over time does not affect the future usability of the resource. For example, Multiplex lines are temporally divisible in the context of a communication activity. A resource is physically divisible if each division of the resource can be used or consumed by an activity. So, if the activity is to 'sit_on' a wooden chair, accordingly the chair can not be considered physically divisible. While, on the other hand, if the activity is to 'fuel_a_fire', then the wooden chair is considered physically divisible.

Quantity**:** A resource point specifies a resource's quantity at a specific time point. For example, there exists a resource point, for resource 'plug' at time point '90', with quantity of '100' units. A resource point can be extended to differentiate quantities at different locations by specifying that the resource point of the resource 'plug' is '125' at time point '90' in location 's2'.

Component**/Structured:** 'Component of' specifies a resource as being a part of another resource implying that a resource consists of one or more sub-resources (i.e sub components). component_of (wire, plug_on_wire) specifies that 'wire' resource is a component of the 'plug_on_wire' resource. A resource is said to be physically structured if it has at least one subcomponent [13].

Source**:** Furthermore, resources may be classified into [13]: ***manufactured resource*****s** that are produced by consuming at least one resource**,** ***bought-in resources*** which are purchased resources, and ***sales resources*** which are saleable resources. Part of the resource taxonomy is presented in figure 6.

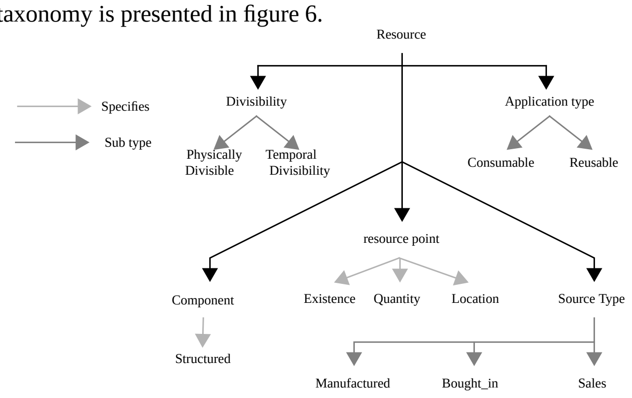


Figure 5. <u>Resource Taxonomy</u>

<u>Semantics and Common Sense Axioms</u>

As discussed earlier, the competence criterion focuses on how well the model supports problem solving. We define a model's level of competence by a set of questions it should be able to answer either directly or through deduction. Following are a subset of questions we have considered in the creation of the TOVE model.

<u>Activities and Causality</u>. ***Conditions***: What is the current status of an activity? What alternatives exist? ***Causality*****:** What conditions have to be satisfied to perform activity? What conditions will be satisfied when the activity has been performed? ***Abstraction*****:** What subactivities can the activity be divided into? What superactivities is an activity part of?

<u>Time</u>. ***Time Point***: When does the activity start? ***Time Period*****:** What is the start time and end time of the activity? ***Time Window*****:** What is the earliest/latest start/end time? ***Time Relation*****:** Is activity 1 before/after/during activity 2?

<u>Resource</u>: ***Existence***: How much of the resource exists at time t? ***Consumption*****:** Is the resource consumed by the activity? If so, how much? ***Divisibility*****:** Can the resource be divided and still be usable? Can two or more activities use the resource at the same time? ***Structure*****:** What are the subparts of resource R? ***Capacity*****:** Can the resource be shared with other activities? ***Location*****:** Where is resource R? ***Commitment*****:** What activities is the resource committed to at time t?

Part of the TOVE model is the representation of terminology definitions in the form of first order logic and its implementation in Prolog. This provides a deductive query capable of answering many common sense questions. The following are an English description of some of the definitions:

- Activity 1 is before Activity 2 if the end time of Activity 1 is less than the start time of Activity2.
- An activity can be executed if its enabling state is enabled
- A non-terminal state is enabled if the boolean combination of its substates are enabled.

- A resource is physically divisible if it has at least one sub-component.
- A resource may be consumable if it is physically divisible in its role in an activity.
- A resource is reusable if it is temporally divisible in its role in an activity.

CONCLUSION

The goal of the TOVE (TOronto Virtual Enterprise) project is to create a data model that: 1) provides a shared terminology for the enterprise that each agent can jointly understand and use, 2) defines the meaning of each term (a.k.a. semantics) in a precise and as unambiguous manner as possible, 3) implements the semantics in a set of axioms that will enable TOVE to automatically deduce the answer to many “common sense” questions about the enterprise, and 4) defines a symbology for depicting a term or the concept constructed thereof in a graphical context. In this paper, we described our terminology for the representation of activities, states, causality, time, abstraction, manifestations, and resources. Of particular concern to us, has been the lack of criteria for measuring whether one enterprise reference model is better than another. Consequently, we introduce a set of criteria, and briefly define the *competence* of our representation in terms of questions it can answer, and the *semantics* of our terminology in first order logic with an implementation in Prolog. The latter provides the capability to answer common sense questions using deduction.

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.

REFERENCES

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

[2] Allen, J.F. Towards a General Theory of Action and Time. ***Artificial Intelligence.*** 23(2):123-154, 1984.

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

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

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

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

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

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

[9] 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.

[10] 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.

[11] 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.

[12] 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.

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

[14] 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.

[15] 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, Purdue University, West Lafayette, IN 47907, 1991.