1999

Object Oriented Enterprise Modeling and Distributed Cognition at a New University: CSU Monterey Bay

Armando A. Arias Jr.
California State University, Monterey Bay, aarias@csumb.edu

Beryl L. Bellman
California State University, Los Angeles and Ptech, Incorporated

Object Oriented Enterprise Modeling and Distributed Cognition at a New University: CSU Monterey Bay

Armando Arias

California State University at Monterey Bay

Armando_arias@monterey.edu

Beryl Bellman

Ptech Incorporated & California State University Los Angeles

Bellman@ptechinc.com

1999

INTRODUCTION: THE ADVANTAGES OF OBJECTS

Object Oriented methodologies have steadily grown since the 1960s when Simula was first developed in Norway, which led to Small Talk and other applications. In 1989 several OO developers formed a consortium known as the Object Management Group or OMG to create OO standards (1) and a unified modeling language. (2)

The OMG defines object management as

> *"…. Software development that models the real world through representation of "objects." These objects are the encapsulation of the attributes, relationships and methods of software identifiable program components. A key benefit of an object-oriented system is its ability to expand in functionality by extending existing components and adding new objects to the system. Object management results in faster application development, easier maintenance, enormous scalability and reusable software (c.f. www.omb.org)".*

Although first used for software applications development, several developers recognized the relevance of Object technology for enterprise process modeling and business reengineering. (3) Among the first was John Edwards in this county and In Sweden, Ivar Jacobson, one of the early developers of Use Case methods in Object Oriented software development (1992), formed Objectory to use OO technology for business process reengineering (1994). Jacobson proposed a congruity between software development and other types of human systems:

Today, object oriented technology is being used very successfully in software and in every type of system built. These systems are comprehensive, understandable, changeable, adaptable and reusable. Because the same approach is used both for realization of software systems (in the code) and for abstract models of the system (system and design models), it is easy to trace properties between the two models…. If the same technique is used to model a business (sic or any activity system) as used to build the supporting information system, the transition between the two activities will be both easy and distinct (p 74, 1994)." (4)

John Edwards invented event modeling (referred to in UML as activity modeling), and introduced dynamic classification and intersection into Object Modeling. Edwards collaborated with James Odell and James Martin in the writing of Object-Oriented Analysis and Design in 1989, which Odell and Martin later published in 1992. (5) Then, Udell and Martin published later extensions of their work (1993 and 1998) that expanded enterprise structures such as business rules, meta modeling, power types, dynamic and multiple classification. (6) In 1994 Oussama Ziade and James Cerrato established Ptech, Inc. to further develop Object Oriented enterprise modeling with the technology that Edwards, Udell and Martin used in their research. Ptech provides modeling capabilities that furnish organizations both a visual and logical integrated blueprint of their enterprises using UML standards and it's a powerful code generation capability. (7)

The shift of emphasis towards enterprise modeling to incorporate business capabilities into the definition of objects beyond those relevant to software development did not significantly change the basic definition of what constitutes an object.

When Jacobson first defined an object in the context of software engineering as

"… An entity able to save a state (information) and which offers a number of operations (behavior) to either examine or affect that state. An object is characterized by a number of operations and a state which remembers the effect of these operations (p 44, 1992)."

He later, after developing interest in business process and reengineering,clarified

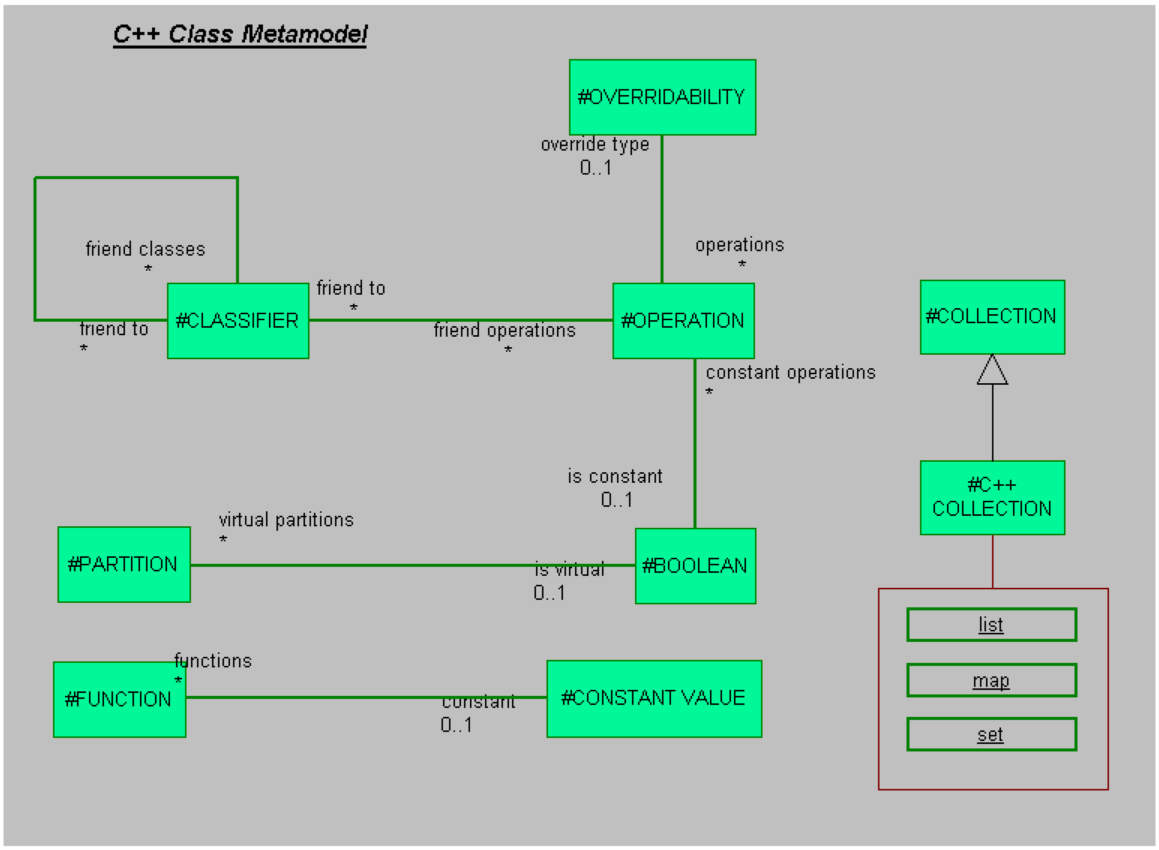
"…. An object is an occurrence that can contain information and offer behavior. Other objects can make use of the object's behavior to obtain or alter information that is attached to the object (p 48, 1994)."

Initially in the domain of software development an object was defined as a character string, queue, file directory, text editor, program, compiler and a computational operation (Firesmith and Eykholt: 1995). The use of OO in business enterprise modeling has expanded this definition so that an object now can be a product, a process, a person, a team, a company, an application or the inter-relationship between other objects. This latter characterization is consistent with objects as used in activity theory. This body of work recently increased its influence by emphasizing cultural and developmental factors in cognition (c.f. Wertsch, 1981, Engestrom and Middleton, 1996, Salomon, 1996, Nardi, 1996 and Cole, 1998). Here social and cultural properties are objects in the same way as physical properties in human environments. Objects are cultural and socially mediated artifacts in communication.

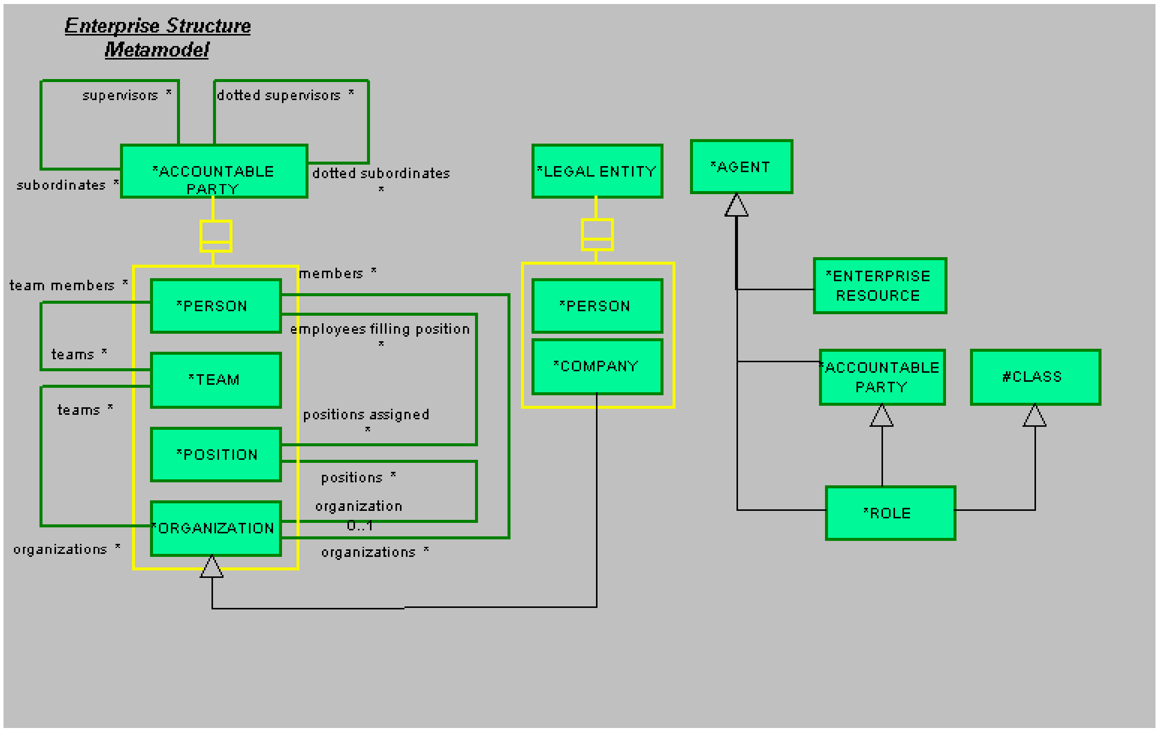
Although Objects stand for concepts, they do not require realistic iconic representations as pictures of what they depict. As Baigrie (1996) emphasizes in the case of scientific illustration, it is "…imperative the image not resemble its object as much as it might," and in an essay about Cartesian uses of scientific illustration, cites Descartes' view:

"…. Engravings represent to us bodies of varying relief and depth on a surface, which is entirely flat. Moreover, in accordance with the rules of perspective they often represent circles by ovals, better than by other circles, squares by rhombuses better than by other squares, and similarly for other shapes. Thus it often happens that in order to be more perfect as an image and to represent an object better, an engraving ought not to resemble it. (P 122)"

The following two diagrams demonstrate that the same object symbols and structure are relevant for both application development with object oriented programming and in Object Enterprise Models. The following is a software Object diagram in Ptech Framework:



And, the next diagram is a metamodel for business objects using the same iconic representations and logical structure:



In OO models, objects are pictured on the computer screen as maps formed by "signs" with their connective relationships. Once a "map" of objects is produced, users are able to navigate and visualize very complex relationships. Objects contain types of data, such as, cost, schedule data, weight and other relevant information. Objects are weighted and derive meaning through their connectivity with other objects at the point of interaction. As evidenced in neural networks, the value of any particular weight has no particular significance, as it is the patterns of weight values in the connected system that produces information. The metamodels presented above represent the underlying logical schema for different types of processes: in the first instance for code generation in C++ and in the latter for modeling organizational structure. In Ptech there are many different metamodel schemas for various types of enterprise activities and states, as well as contains the ability to adapt or create new metamodels in accordance with any organizational unique requirements.

Another important property of an Object in dynamic OO models is its ability to perform work in "methods." A method is any "hidden implementation of an associated operation (Firesmith and Eykholt, p251). An Object can be given the interactive capability to perform functions, such as, performing computations, gathering data from other computers, showing video of servicing a part or accessing a 3D-CAD drawing for viewing. This "active model" is much more than a map for navigation in an abstract process-model. It becomes the actual work environment for individuals and teams. It creates an occasioned environment for learning, assessing issues and impacts, communication, configuration management and control and more. In short, it is the user interface or " control center" from where to manage the business.

Although standardization of objects prompted a growing number of companies to develop a range of OO technologies there are contrasting approaches that contain unique logical approaches to problems and modeling methods. These differences are significant for the emergence of different forms of distributed collaborative work and distributed cognition (Arias, 2000).

In this paper we consider the logic of objects and the kinds of distributed cognition or activity systems they generate in different types of implementation and application environments. We provide examples for types of object logic and ontologies used for software application development, business and technology enterprise modeling, knowledge management and CSCW (computer supported collaborative work). We contrast OO enterprise modeling technologies to display their methods, use-cases and patterns relevant to these different implementation environments. In this manner we examine of logic of these technologies as distributed cognitive systems.

DIAGRAMMATIC REASONING WITH OBJECTS

Several years ago one of the authors of this paper (Armando Arias) provided another of its authors (Beryl Bellman) with an epiphany-type experience. A third colleague was rehearsing a presentation he was going to give at a customer briefing, and was showing a slide from a text showing an object model with text under. Armando asked Beryl what he was looking at, and Beryl replied he was reading the text and would then look at the diagram to support or re-emphasize what was stated in the narrative. Armando commented that he was instead looking at the objects and, only if necessary, would later read the supporting text. He then made the point that the object model in this case was not supporting the text, but rather the text was attempting to explain in a much weaker sense the model.

This prejudice of text over image is an unfortunate by-product of Western literacy. Kress and van Leeuwen (1998) describe how this prejudice develops:

> *"In the early years of schooling, children are constantly encouraged to produce images, and to illustrate their written work. Teachers comment on these illustrations as much as they do on the written part of the text…. Unlike writing, illustrations are not corrected or subjected to detailed criticism…. They are seen as self-expression, rather than as communication…. By the time children are beyond their first two years of secondary schooling, illustrations have largely disappeared – form the children's own texts as well as from the texts produced for them. Whereas texts produced for the early years of schooling are richly illustrated, later on visual images give way to … greater proportion of verbal, written text. In as much as visual images continue they have become maps, diagrams or representations with a technical function (p 15)."*

Yet today there is a large and growing literature on centrality of visual images, particularly concerning diagrammatic reasoning of which OO models are part. This literature address how the use of diagrams and other forms of imagery, including for our purposes objects, are opposed to linguistic and algebraic forms of representation. In this form they provide unique insights about human cognition. Larkin and Simon (1995) describe how research on visual and diagrammatic reasoning has centered on the features that are also highly relevant to OO modeling;

- Diagrams group together all information that is used together, thus avoiding large amounts of search for the elements needed to make a problem-solving inference.

- Diagrams typically use location to group information about a single element, avoiding the need to match symbolic labels.

- Diagrams automatically support a large number of perceptual inferences, which are extremely easy for humans.

In the preface to their comprehensive text on diagrammatic reasoning, Glasgow, Narayanan and Chandrasekaran (1995) provide an interesting contrast of some cognitive operations when we confront visual problems that can provide us insight into how Objects are construed.

They cite two examples respectively drawn from Lindsay and Pylyshyn:

> *Take one-step north, one step east, and one step south. How do you get back to the starting point?*

And

> *Image a vat of blue paint, and imagine pouring yellow paint into it and stirring it. What do you see happening to the paint in the blue vat?*

In the first instance we follow each step as Objects linked together into a process flow, and then reverse directions in an opposite flow back to the start. In the latter we know things about the characteristics of the color paints in the respective vats, and have a formula learned early in pre-school art classes about mixes. Here knowledge is coterminous with the presentation of the Objects. Both reveal system of relationships. In the first, direction Objects gain meaning according to their position relative to other direction. Objects in the instruction. In the second, culturally grounded knowledge Objects (cultural forms of color differentiation and terminological systems) are related to information Objects (consequences of mixing or interrelating color objects) in vats (as another type of objects with properties) that interrelate to chemical Objects (the paints they can contain).

The Objects in OO models are compatible with distributed cognitive systems as they entail a neural network mapping of relationships between objects. This gives a distributed representation rather a simple focus on Objects. The elements in the models have no representational meaning by themselves. Instead, meaning is located in the patterns of relationships formed through object interactions with other Objects.

The meanings of OO models are derived from ongoing interactions between components of the system, including the relationships and perspectives of those that design and utilize them. Such representations must be multi-dimensional and visual in character. Textual formulations of such models tend to be Garfinkelian "poor relatives" or weaker versions rather than elaboration.

ENTERPRISE MODELING METHODOLOGIES

Enterprise modeling is grounded in systems thinking. A model's value is derived according to use. Models are both political and adversarial as they promote or follow policy and organizational strategy. As Morecroft and Sterman (1994) describe

> *"The modeler is, in part, a facilitator, one who designs and leads group processes to capture team knowledge. The modeler designs and delivers learning laboratories that embed models in an overall learning context of group dialogue and experimentation (p xviii)"*

There is no formulaic design in modeling as modelers must be open to what there is to be discovered. Instead, modelling involves more what Peter Checkland (1981) calls a "guide to action." Such a guide, he advises, must be neither vague like a high level vision or general philosophy nor precise like a technique or method that does not permit creative innovation. Instead, methodology is a "science of procedures" and a set of principles of method which in any particular situation have to be reduced to a method uniquely suitable to that particular situation (pp. 162-163)." Checkland's approach to systems thinking and modeling has had a significant influence on many enterprise modelers, and was re-popularized by many in the field as providing a framework for building multiple perspective models. For that reason we discuss it some detail below.

Checkland differentiates hard and soft systems. The latter are complex and entail human activity, which are not fully predictable as the systemic or hard systems logic in the physical sciences. Soft systems are highly adaptable and emergent, and so require a distinct approach or guide to action. He outlines seven stages in his soft systems methodology. The methodology involves two types of activity – those that are real-world involving people within the problem situation, and "systems thinking" activities. The first two stages are real world. They are not "the problem," but instead an attempt to capture the situation where the problem is perceived to exist. Stage one involves recording the relevant objects in the situation that includes both what are part of the "slow-to-change" structure and those of the "continuously-changing process." The second stage is building relationships between objects to view how "…structure and process relate to each other within the situation being investigated." Thus, in stage one the problem is unstructured whereby relevant objects are located and defined, and in stage two relationships are specified and built. According to Checkland the function of these two stages is "…to display the situation so that the range of possible … relevant choices can be revealed.

Stages three and four entail "systems thinking," apart from the situation. In stage three, one attempts to define what Checkland calls a "root definition" of the underlying systems that are causing the problem. Such root definitions are explicit formulations of what these underlying systems are, rather than what they do. These definitions must be psychologically real or salient for the client. Checkland uses the mnemonic CATWOE to underscore what this involves. The definitions must be real to the customers, the system's actors, its transformation process, the *Weltanschauung* or world view of the corporate culture – which generates organizational meaning, and be consistent with the environmental constraints to which it is subject.

Once the root definitions are formulated and consistent with CATWOE criteria, conceptual models can be built. Stage four then consists of building models of the human activity systems relevant to the root definitions. This stage involves two parts: first, is to create a general model of any human activity system to serve as a benchmark for comparison, and the second entails transforming the model into a form that is suitable for the problem being confronted. Thus, we might use the "balanced score card," "Zachman" or other informed metamodel depending both on the nature of the root problem and the organizational cultural preferences of a client. (7)

In stage five the models are brought into the real world to see from the client's view how they fit with the situation. This is meant to spark discussion and debate among members of the client's team, and to insure that the appropriate organizational view or views have been represented and captured. The debate leads to an iterative process whereby the model is refined and reintroduced for further comment. In stage six an action plan is developed from the comparisons at stage five. This plan must meet two criteria; the changes must be both desirable and feasible given the organizational culture, power structure and the history of the problem situation. In stage seven direct action is taken to improve the problem situation as based on the models.

Recently in a Harvard Business Review essay on corporate knowledge management strategies, Hansen, Nohria and Teirney (HBR, March-April, 1999, pp. 106-118) distinguished two basic and contrasting methodologies used in the major management-consulting firms. The first involves a "codification strategy" where knowledge is classified, coded and stored in knowledge bases as reusable, plug and play methods to adapt in different types of customer engagements. They cite companies such as Anderson and Ernst & Young where knowledge is codified using a "people-to-documents" approach. Information is extracted from the engagement where the knowledge was first developed, sanitized for re-reuse and then codified for retrieval in other engagements. Such codified knowledge includes interview guides, work schedules and benchmark data, market segment analyses and the like. They cite the example of designing an enterprise resource planning system where such codified knowledge greatly reduced project life cycle time and costs. The second approach, in contrast, entails highly customized solutions to complex problems. Here knowledge is distributed through person-to-person contacts, and computers are used to facilitate communication rather than for information storage and retrieval. The authors refer to this as a personalization strategy as used by strategy consulting firms such as McKinsey, BCG and Bain, as well as the cross-functional legal and economic consulting practices of PriceWaterhouseCoopers. The first approach is less expensive and costs on the average of $600 a day, while the latter being more creative and requiring a high level of individual consultant expertise, averages around $2000 per day.

This difference also characterizes the contrast between recipe and strategic forms of consulting. In recipe consulting problems are viewed according to established formula and defined tools. In this instance the methodology and tool are considered superior to the expertise of the consultants working on a problem. That is, the "tools" are considered indispensable whereas the consultant is readily interchangeable. New consultants are trained in tool use and established procedures, and are dissuaded from thinking "out of the box." In strategic consulting, expertise and knowledge are the primary focus. Tools are meant to augment the expert consultant's work, not to pre-determine solutions.

Many consulting groups fail to recognize and distinguish between a situation requiring strategic consulting versus a plug and play set of methods. Hansen, Nohria and Teirney even argue that the two approaches are incompatible, and that consulting organizations must determine which type of practice they have. The type of dilemma resulting from incorrectly assessing whether to use a plug and play or the soft systems methodology can lead to enormous waste for both the client and the consulting organization. Canned solutions get offered and bought into, but are never really implemented in the way that was originally presented to the customer. This results in unused binderware sitting on management's shelves at the best, and an irate, dissatisfied customer at the minimum of worst.

One example we analyzed was in a major middleware professional services consulting practice in a large computer and services corporation. The professional service organization wanted to articulate methods for doing successful middleware architecture consulting. The original assumption was consistent with the corporation's commitment to the knowledge codification methodology, where methods take precedence over the individual knowledge and expertise of consultants. One of key method is their information technology planning method, which they defined as a suite of tools that document, maintain and communicate architecture plans. The role of the consultant is that of a tool user "… to assist in the information collection, assembly and modeling of the architecture and transfer knowledge of the process to client representatives."

When applied to middleware, the Professional Services methodologists argued that each consultant engages in a pre-specified formal method. These are of composed of six activity stages: (1) develop statement of work, (2) initiate project, (3) perform requirements analysis, (4) design the solution, (5) develop implementation plan and (6) close out or conclude the engagement. The consultant had at hand a set of formal documents, power point presentations, work sheets, tables, customer-sign off memo templates and checklists. The general PS organization collected these "tools" and created a CD-ROM disk set that they called a workbench. Although it would seem that this CD is a step towards improving the codification methodology, in practice most consultants objected to using it. Some objected stating it just got in the way, and that once they understood how to utilize the corporate approach, they did not need a workbench

Another group of consultants was those who were primarily concerned with activities 4 and 5 – which entailed creative and often novel solutions to a customer's problem. Although many were trained in doing middleware architecture design, less than a dozen across the entire corporation were recognized as truly competent. These "guru consultants" (as they were called) objected to the idea that their approach to a solution could be codified and easily re-used, as each design required an understanding of the individual customer, their organizational culture and IT scalable environment. The corporation nevertheless, wanted to codify the "best practices" and organized an international meeting to determine processes for best design.

After many days of "woods" meetings these "guru consultants" finally agreed on a set of processes that were more presented as conference deliverables than as real process recommendations. Later when we had the opportunity of discussing the meeting with several who participated, they continued to object to the model they agreed upon, stating that each consultant has his/her own approach. To specify these processes would be like "writing the exact processes for how Piccasso painted … you just can't specify the details of creativity." The underlying issue or "root definition" of the problem situation was the disjunction between re-usable methods and the need to develop methods that emerge from a deep understanding of a customer's problem.

Another way of distinguishing these methodologies is to recognize the distinction between complicated and complexity. Complicated systems can have many different components and perform sophisticated tasks, but can be accurately analyzed and fully understood. Complex systems contain non-linear relationships and feedback loops, dynamically change over time and exhibit emergence of new and unpredictable objects, patterns and relationships. Human activity systems or organizations are complex and adaptable (Cilliers, 1998 Mainzer, 1994). A single method or set of methods cannot be defined. However, "prefabricated" methods may work in limited domain complicated environments such as implementing a resource planning system as a reusable design. This does not entail creating a solution to a complex organizational problem, rather it is the installation of an approach that already has a predefined fit. This is the bread and butter of companies such as Anderson and Ernst & Young, who hire and train a large cadre of entry level, low cost consultants.

The design of a middleware solution requires the creative solving of a problem, which in turn requires a real understanding of the problem situation. In this case knowledge is not contained in the Objects representing processes, but in the creative connectivity of Objects in pursuit of a solution. Although the corporation had its methodology and tool-kit to analyze a customer's business and IT requirements, the outcome of their approach was predetermined and did not seriously address the customer's problem situation. The middleware design architecture consultants seldom even referred to that work, and so it was mainly utilized in pre-sales rather than to inform anyone about how to deal with the middleware design problem.

In adapting a soft systems methodology as an action guide, an enterprise modeler more effectively captures business requirements to resolve the problems in customer engagements. Rather than using prefabricated designs, this type of consulting creatively addresses the customer's problems situation as it exists in the real world though connectivity of Objects in the course of real problems.. To do so involves capturing the objects and relationships comprising that situation, and then uncover the root problem definition to build models that generate viable solutions. Instead of applying an "out of the box" codified "people-to-documents" approach, it is possible to inform IT architects and data modelers by generating from Object relationships the knowledge they need to design solutions.

The soft systems methodology pertains to any environment where there is a real enterprise or organizational problem to resolve. We recognize that in the same manner as in being a great surgeon, master chef or expert at any trade having the right tool is critical to success. Possessing knowledge without the right tool results in *bricoleur* or ad hoc construction. The bricoleur uses whatever is at hand to make a fix. The results are tenuous and temporary. However, in using the right tool, the expert is able to build the most robust solution possible.

OO enterprise business modeling tools entail a portfolio of metamodels as underlying logical schemas informed by management, organizational and communications theories that are tailorable to fit the specifications of any customer engagement. These tools greatly facilitate the capture of information, and the portfolio of informed metamodels greatly assets conceptual modeling, and permits easy modifications in the iteration phases. In this manner, OO enterprise modeling involves both the talents of expert consultants as well as having a powerful modeling technology to resolve the most complex enterprise problem situation.

SECTION II – DEMONSTRATIONS OF OBJECT ORIENTED ENTERPRISE MODELING

OBJECT ONTOLOGIES AND STANDARDS: AN E-COMMERCE EXAMPLE OF CSCW

The use of OO logic in models is also being applied in work on Web systems development for electronic commerce and other distributed programs. These efforts require the formulation of ontologies and standards across industries and national boundaries. The types of knowledge about standards being explored today are the subject of many different distributed collaborative efforts. Such initiatives as RosettaNet, CommerceNet, Ontology and OBI (Open Buying on the Internet) involve collaborative efforts to define the ontology of objects and Internet standards. (8)

Because of the enormous complexity of the distributed work involved, most of these initiatives are seldom integrated and often overlap and replicate each other's efforts. To be effective this distributed knowledge work requires collaboration across the other related standards groups for constructing a viable framework. The effort entails collaboration on the nature of the objects involved. This task entails what Brian Smith (1996) describes as registration of phenomena into Objects. Object modeling connects differing systems objects with other types of systems and their respective objects in a common framework environment. Such a framework must be relevant to all of major electronic commerce areas: business-to-business commerce, business-to-consumer commerce, enterprise computing, education and research.

An object enterprise model framework is dynamic by adapting to environmental changes and enabling users to ask “what if” and other types of enterprise specific questions to measure the systemic effects of different future scenarios from different views of electronic commerce. It provides for different types of queries about both the as-is framework and the potential and/or real implications of the introduction of new technologies, effects of new or changes in economic and governmental policies and corporate strategies and emerging markets.

Each of the electronic standards initiative projects entail the organization, distillation and collective action on information among global businesses, many of whom are in market competition with one another. In order to facilitate work it is necessary to obtain information that on the one hand protects corporate knowledge assets while distilling the information into a form that enables collective discussion, agreement and eventual action. This involves creating processes to link diverse and complex data structures and knowledge processes, as well as collating and interpreting them into a framework for collaborative and collective action. It entails working with proprietary information in such a manner that corporate rights are maintained while transforming and disseminating knowledge across industries and discipline domains.

Recently, Ptech developed a dynamic enterprise model of the World Wide Web Consortium (W3C)- giving stakeholders’ web accessible roadmaps. The model was a prototype demonstration designed according different views that relate a particular group’s participation to each of the other domains, activities and groups in the consortium. It offers each group a resource to better conduct distributed collaborative work and to align with other relevant initiatives both internal to and external to the W3C.

Ptech built the prototype Ptech demonstration model of the W3C Roadmap based on the model developed by one of the original the founders of WWW, Tim Berners-Lee. Ptech’s version of his model provides both a visual and logical representation that is easy to navigate and permits a number of queries relevant to program management and strategic alignment of all model components and objects. Ptech provided what they refer to as a “knowledge control panel” that provides connectivity to other internal and external programs, and that can facilitate the work of each type of group in the W3C. Ptech recognizes that different people need to communicate ideas according to their role and perspective in the organization. With the model such different views are available to enable full system integration, viable knowledge networking and effective distributed collaborative work.

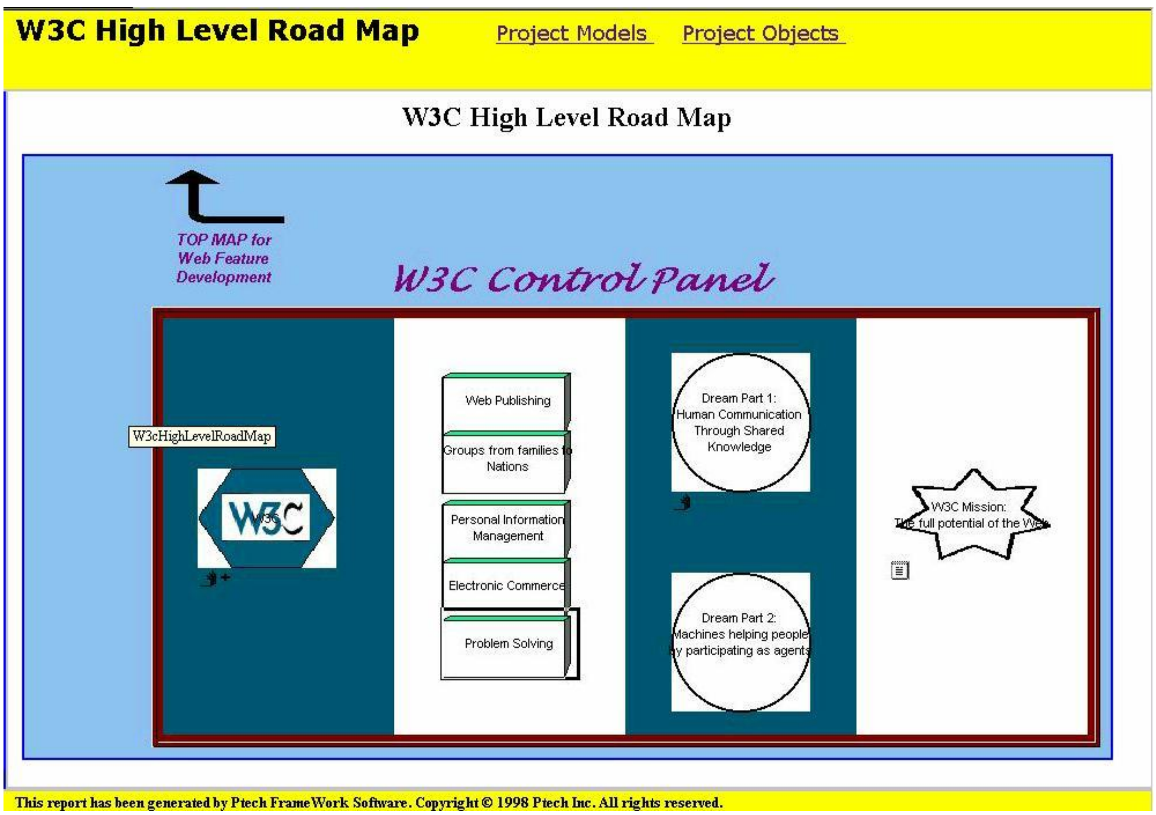
The model enables stakeholders in the W3C to quickly understand the impact of their respective contributions by providing traceability from vision to implementation, and displaying relationships between group missions, projects, activities and deliverables.

Ptech's Framework architecture is driven from its corporate vision of providing an "adaptive enterprise knowledge control panel" - one which can adapt quickly to a company's requirements and one which can enable companies to more rapidly respond to changes in their environment. An effective control panel for the enterprise is analogous to a control panel in the cockpit of an airplane. The flight deck control panel provides constant feedback to the pilot on the changes and status of the internal workings of the plane and on the changes and status outside the plane in the external environment. The pilot has a goal to go from one location to another and requires the control panel to quickly inform of significant changes in both the environment and the key functions. In the pilot's world "quick learning" must be followed by "quick response to change". The architecture of the plane must be clearly understood internally and relative to the plane's external environment before a control panel can provide meaningful information. The same holds true in the business world: before a meaningful "knowledge control panel" can be put in place, the 'enterprise architecture of the company must be explicitly blueprinted and understood'.

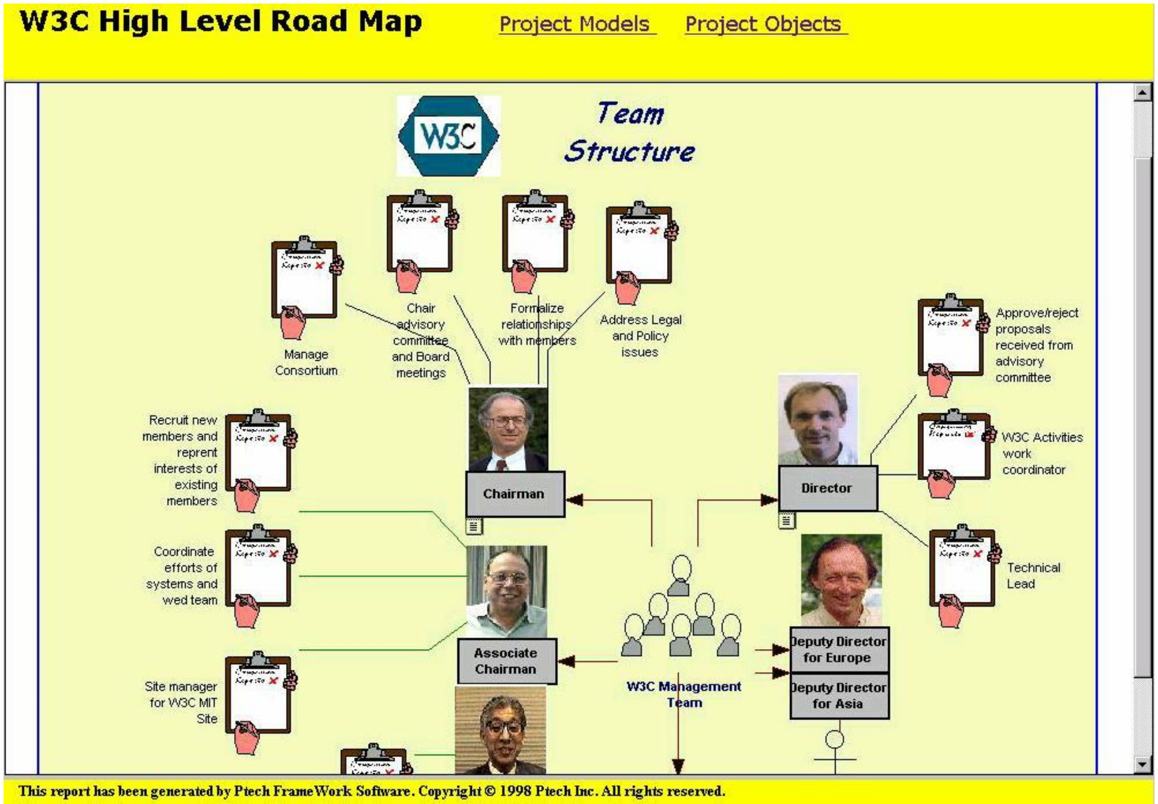
The Ptech framework technology allows organizations to quickly capture such visual and logical architectures of enterprises using Object logic and a tailorable set of metamodels. The integrated set of object-oriented methods and tools enables the modeling, modifying and instantiating of strategic intent, enterprise or business requirements, technical requirements, organization structures, roles and responsibilities, interrelated applications incorporating best practices and re-usable business objects. This technology is also able to create html with graphical "hot spots" for navigation to other related knowledge tools, as well as generate software code directly from the business models (C++, corba, Java, XML oracle8, Forte, etc.)

In the prototype Ptech restructured Tim Berners-Lee's road map model using several levels of representation. At the top level they reduced the number of objects to eight, each having sub architectural and process diagrams.

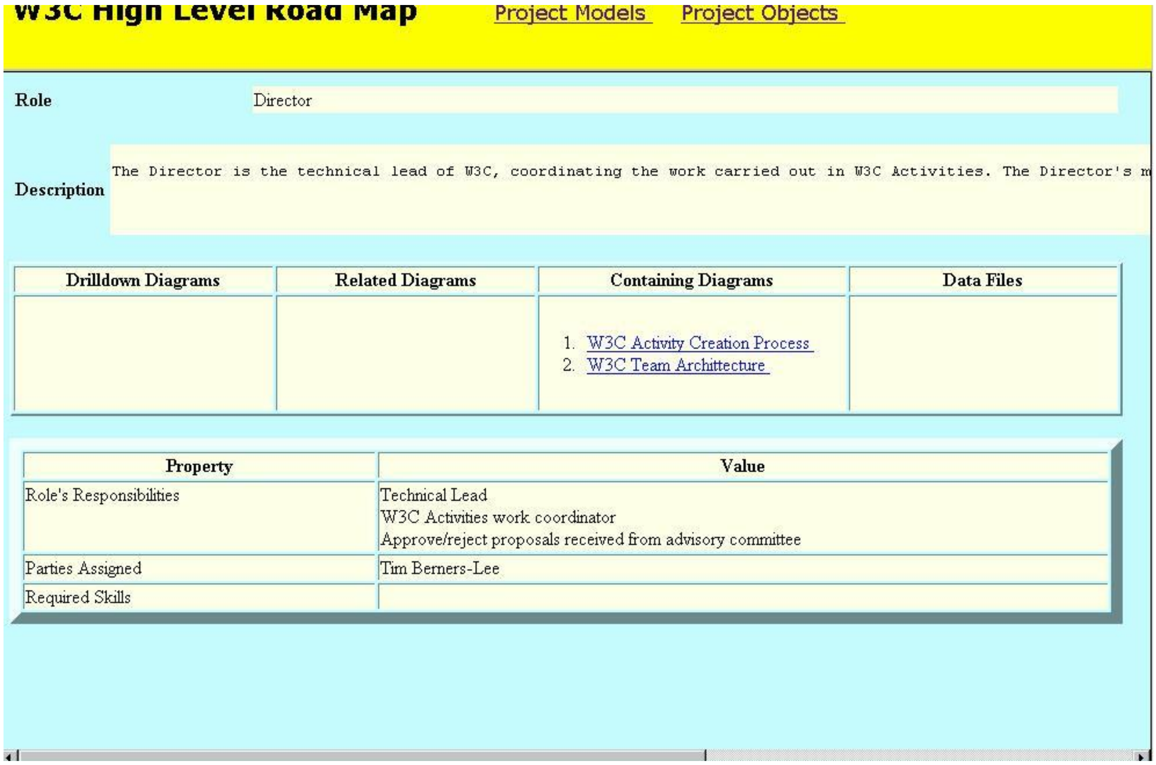
The following is a snapshot showing the upper level representation of our road map control panel as presented in HTML format accessed using a browser:



Each of these objects nine objects contained forms and views describing the object and its relationship to other objects that are opened by double clicking on them. For example, by clicking on the W3C icon object the "team view" is opened:



This same view can also be brought up as a form showing object definitions and internal relationships:

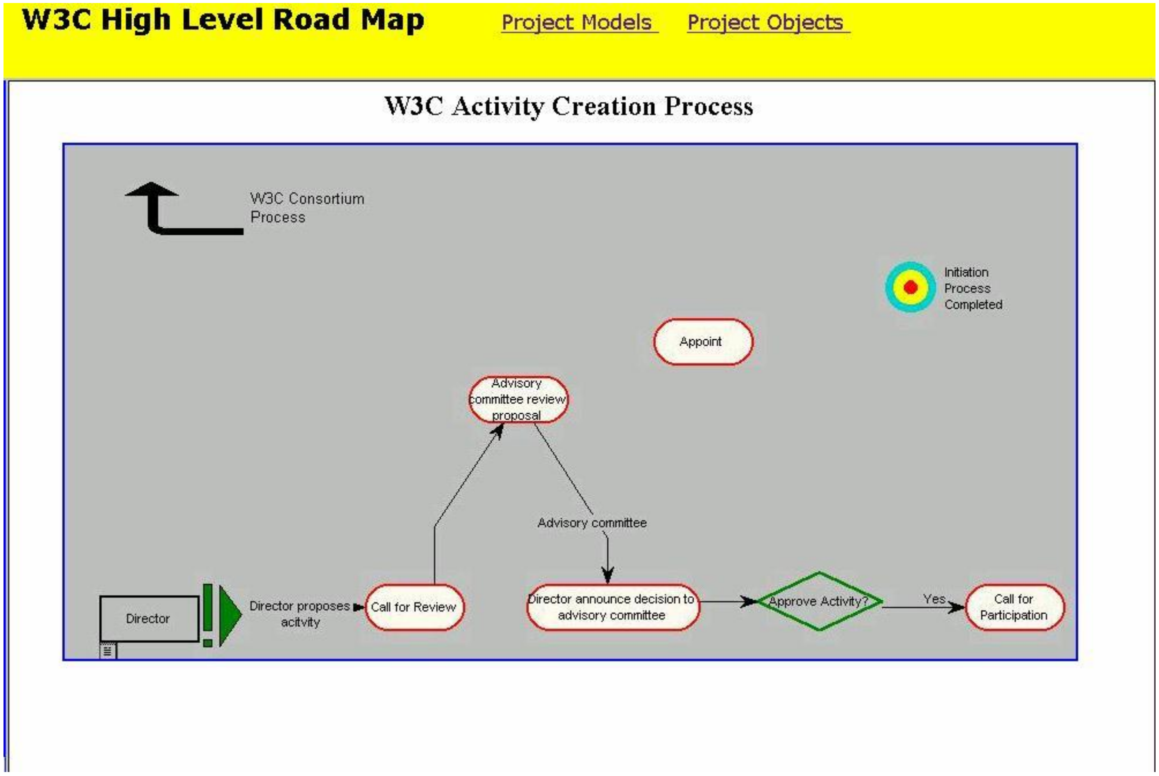

W3C High Level Road Map   Project Models   Project Objects

Role	Director
Description	The Director is the technical lead of W3C, coordinating the work carried out in W3C Activities. The Director's m

Drilldown Diagrams	Related Diagrams	Containing Diagrams	Data Files
		1. W3C Activity Creation Process 2. W3C Team Archittecture	

Property	Value
Role's Responsibilities	Technical Lead W3C Activities work coordinator Approve/reject proposals received from advisory committee
Parties Assigned	Tim Berners-Lee
Required Skills	

Also, each of the objects in these representations can be opened to show a range of types of sub-diagrams containing workflows, business and/or technology architectures, processes, process steps and more. The following is one such diagram showing an activity creation process.



Again, each of these objects can be opened to reveal internal process steps, architectures, and more.

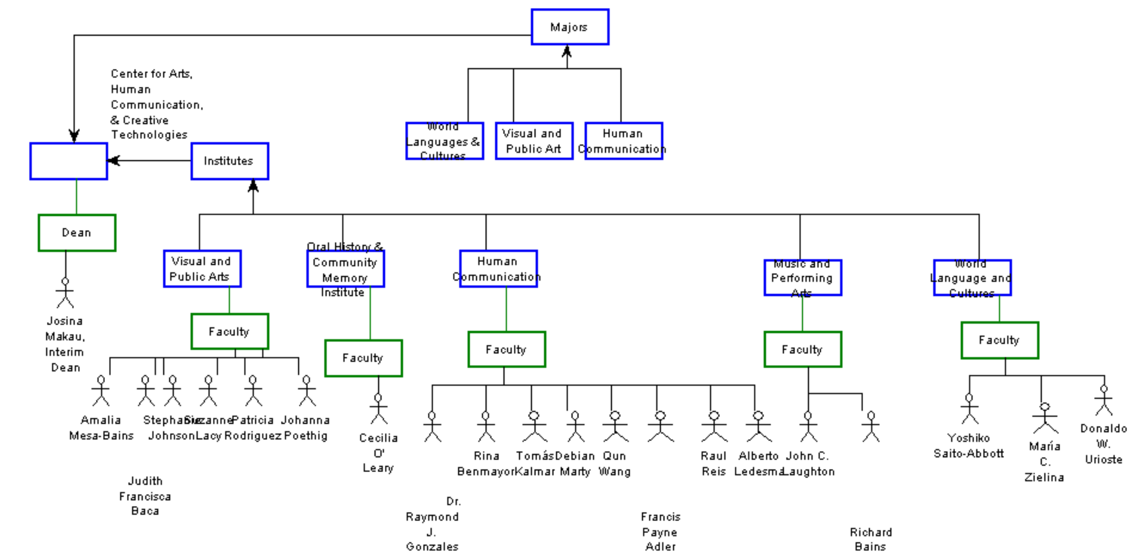
These OO frameworks focus on both business and IT architectures to effectively model the processes and dynamics of distributed Intelligence. In this manner they address the computational aspects of distributed intelligence by optimizing through dynamic models task allocation, interactions between work groups, activity groups, consortia and other organizations addressing standards and processes relevant to electronic commence and the Internet. This defines and improves group processes and facilitates organizational representation and collective learning. It also provides for more consistent management of numerous efforts by strategically aligning their contributions and work, and provides for more effective group cognition. In this way, OO enterprise models are cognitive tools that can team individuals through computer based technologies to form a joint intelligence that shares and distributes labor during the group processes.

CREATING A NEW UNIVERSITY USING OBJECT ORIENTED ENTERPRISE MODELING

OO enterprise modeling has been effectively used for the creation of a new university and development of its programs. An enterprise model was used to assist in handling the transformation of a large portion of a military base Fort Ord, into a state university – the California State University at Monterey Bay. This involved the re-engineering of a massive infrastructure, political negotiations about thousands of acres of "prime real estate," munitions searches, toxic waste clean-up, physical plant remodeling, environmental studies, and much more. The university planners found themselves in the difficult position of having to "brand-new" and "renew" at the same time. The challenge was how to document activities and provide a decision framework for designing programs at both the administrative and academic levels.

The campus decided to use OO enterprise modeling for strategic planning and systems design. The enterprise model provided a way to communicate campus goals and for reaching consensus on how to get there.

The following is one of the many diagrams from that model showing the Center for Arts, Human Communication and Creative Technologies:



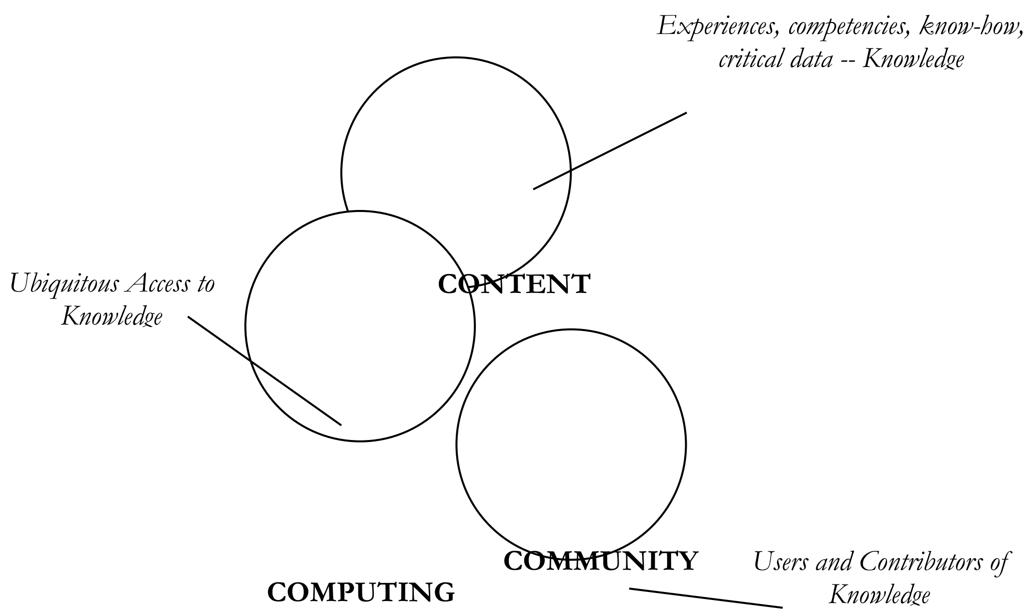
The model was put into use to deal with unexpected consequences in introducing new system-wide IT systems and applications. While setting up training modules, the outside software consultants were not shy in insisting to the California State University Monterey Bay administration they need fifty per cent of key individuals' time for more than a year to implement specified modules. The model, although not fully resolving this issue, has significantly reduced the learning curve of many key faculty and staff by providing them with a systems view of what the new modules entail.

Although this work remains a "work in progress," the campus continues to collaborate with corporate partners in implementing OO modeling approaches and strategies in a manner that leads to advances in on-line teaching, learning, research and assessment. This effort provides unique insights about pedagogy and collaborative research, and enables continual renewal of innovative curricula and learning experiences. The planners at California State University, Monterey Bay take their Vision Statement seriously, and designed their enterprise model around its founding principles - to include "relational" webs to people, activities, plans, policies, academic programs and requirements and space. They regularly query their model to measure potential impacts that new policies, technologies, programs and faculty personnel will have on academic and student affairs. For example, when new employees join the Social and Behavioral Sciences Center they are shown their unit's perspective of within model as a way of orienting them to their work unit, and to show how their position interfaces with on other parts of the campus community.

California State University at Monterey Bay is still in the midst of planning and implementing its programs. In response to both student needs and external deadlines, this process involves a broad array of university faculty, administrators, staff, and students, as well as external stakeholders in the community and state, working on a variety of educational planning tasks in a fast-paced, rapidly expanding, and highly decentralized fashion. The use of the campus Object enterprise model facilitates their strategic alignment, allowing planners to design new business processes from vision to deployment. Their vision for the campus model is allows users to perform everything from a virtual fly-by of the university to navigating academic programs and how they relate to major learning outcomes, university learning requirements, student assessment, individual learning plans, student capstones, outcomes-based assessment and career planning.

KNOWLEDGE OBJECTS AND ARCHITECTURE

In their paper, Charnell Havens and Ellen Knapp, speak about Content, Community, and Computing, and make a compelling case about these enablers effectively making the Knowledge Management framework of a typical enterprise. The three-C model collectively address how organizational knowledge is identified, captured, organized, and transferred for use/application by individuals and teams through out the firm.



As a broad term, content conjures numerous interpretation and mental models to different individuals. For an organization, content represents the explicit and tacit information used in the planning, operating, and management of its business. The form and type of content can wary from hard-copy documents, digitally stored data, sound, image, multi-media, and full-motion video, to virtual reality.

In most organizations, content and information systems are dispersed across geography and duplicated between product/service lines. An Object Oriented approach to this situation, can help a Knowledge Management team to take control of the content, rationalize the differences, and manage it to ensure ongoing integrity and intuitive access.

Such a challenging and daunting task, of course, requires sustaining leadership support and a highly disciplined approach towards content organization and management. The scope of content and the design of an OO-based Knowledge Architecture must be guided by the *5 Rights principles:*

- Provide the Right Information to
- To the Right People
- At the Right Time
- In the Right Place
- At the Right Cost

KNOWLEDGE ARCHITECTURE AND KEY COMPONENTS

Knowledge Architecture is a schematic representation of content organization, storage, and indexing/cataloging.

Cataloging **(Content Model)**	*Cataloging is an indexing mechanism that provides an abbreviated set of information and ready access to an underlying object or pattern.* ko-n kp-n
Knowledge **Patterns** **(KP)** **(Information Model)**	*KP represents, proven methods, best practices, consistency of application, KO sequencing and dependency based on organization's policies and risk management.* KO-1 KO-2 KO-n ....
Knowledge **Objects** **(KO)** **(Data Model)**	*KOs are stand alone, tightly bound pieces of content/knowledge that can be reused and readily incorporated in a dynamic sequence of work activities.* KO-1 KO-2 KO-3 .... KO-n KO-n

Knowledge Management must be carried out at all levels of the Knowledge Architecture, by clearly establishing the needed disciplines in the usage of content. It must specifically address the policies, business rules, and security governing the access and manipulation of content.

Object Logic and Distributed Cognition

The logic of objects entails a hierarchy of classes of which one class is the class /Object/. Each class has instantiations that are Objects, and a name that distinguishes it from other classes along with attributes that describe the range of values that a property holds. Objects contain both data and operations. An operation is anything that can be done to an object that is also shared by all other objects of the class. There are also subtypes and super types or further levels of class differentiation. An Object inherits properties or attributes for all higher order classes to which it belongs.

This same logical structure also characterizes terminological systems associated with cultural systems of classification and cognitive processing. Cognitive anthropologist who defined methods for understanding different cultural cognitive and semantic systems using folk-taxonomic elicitation procedures (c.f. D'Andrade: 1995) developed methodologies for generating these from field informants. Folk taxonomies were elicited and pursued from levels of higher order classification exploring the "structure of semantic domains." The underlying theory was that some ontological distinction was made to differentiate each level of contrast. The set of all terms in the structure inherited from the levels of contrast above them, in the same manner that objects inherit attributes from classes in Object theory.

As cognitive anthropology developed, concepts such as story grammars, schema, prototypes, and connectionist networks were identified as the underlying mechanisms in folk reasoning. Object theory likewise locates meaning or knowledge in the connections between objects rather than as characteristics solely in the Objects themselves. Meaning is the consequence of the connectivity of Objects through their situated, distributive structures.

In addition, the metamodels that underlie Object models are more than just modeling templates. Instead, they are the underlying logic or cognitive schema and prototypes that generate the structures of actual enterprise models. Thus, the relation between Object logic in OOP and OO Enterprise Models and Distributive Cognition is not simply metaphoric. Rather, they are structurally identical, and as such are consistent with and compatible to our understanding of distributed cognition and activity systems.

FOOTNOTES

1) The first initiative of OMG was to develop CORBA (The Common Object Request Broker Architecture), which later became central in efforts to establish common a modeling standard. CORBA with its Interface Definition Language (IDL) and Application Programming Interfaces (API) allows applications to communicate with each other irrespective of location, operating system and design through an Object Request Broker (ORB).

2) In 1994 several developers created a unified standard called UML or the Unified Modeling Language to coordinate the capabilities and enable object management across the growing number of OO technologies being created. Various major software development corporations soon joined this effort and expanded to include the OMG group. They soon published additional versions of the UML standard, including the recent release of 1.3.

3) In Norway during the mid 1980s some developers became interested in the Object modeling of enterprises using a IDEF-type form of representation capturing process inputs, outputs, mechanisms and enablers. This effort was conceived to enable organizations to visually represent and connect business objectives and processes.

4) They also published a series of essays in the Journal of Object Oriented Programming and other sources.

5) In 1994 Ivar Jacobson and another early leading OO developer, James Rumbaugh, joined Grady Booch at Rational software to coordinate their respective approaches to co-author the first version of the Unified Modeling Language or UML in 1995. Ptech, taking an enterprise outlook, developed an alternative proposal to their UML. Their proposal gained strong support, and was soon incorporated by the OMG into UML as UML 1.1

6) The OOP focus on UML led those in Rational to shift focus away from business process modeling back to software development and code generation based on UML requirement models. Rational still defines as its primary goal the use of UML to help customers develop or deploy software. However, Ptech maintained a dual focus on UML for both application development and OO business enterprise modeling. They integrated the work of Edwards, Odell, Martin, UML, Zachman, and other approaches into a Meta Framework Model that provides the schema for specific Meta model development. . Their approach is to build a flexible tool with a very dynamic Meta model so that methodology can evolve on the fly.

7) Ziade and Cerrato expanded their software's ability to create enterprise metamodels for business process modeling and engineering making their Framework product viable from "concept to code." Ptech Framework now combines strategic planning, enterprise modeling, business process engineering, Object Oriented Analysis and Design (OOAD) and workflow delivering integrated business and IT solutions in a single environment.

8) John Zachman authored his Zachman framework methodology in the 1980s, which has since become a benchmark for the design of IT architectures. The import of this methodology is to incorporate into the analysis all types of requirements in addition and far beyond technological. For a recent presentation of this framework see Inmon, Zachman and Gelger (1997).

9) These include CBL or Common Business Library as sets of XML DTDs, Channel Definition Formats, Electronic Data Interchange, ICE or Information Content and Exchange protocols, KQML or Knowledge Query and Manipulation Language, XML Namespaces, OBI or Open Buying on the Internet standards, OFX or Open Financial Exchange, OTP or Open Trading Protocol, Process Specification and Process Interchange Languages, Simple Workflow Access Protocols, Universal Data Element Frameworks, Web Interface Definition Languages, XMI Metadata interchange, and more.

REFERENCES

Arias, Armando (2000) "Agile Learning and Technology Infusement at a New University" acceoted for publication in the monograph series by the Julian Samora Research Institute, Michigan State University, 2000

Arias, Armando (1999) " Creating A New University Through Object Oriented Enterprise Modeling," Proceedings of the 1999 Meeting of the International Society of Systems Sciences, 1999

Baigrie, Brian (1995) "Descartes Scientific Illustrations" in Baigrie (ed.) Picturing Knowledge: Historical and Philosophical Problems Concerning the Use of Art in Science, University of Toronto Press, 1995

Booch, Grady, Rumbaugh, James and Jacobson, Ivar (1999) *The Unified Modeling Language User Guide*, Addison-Wesley, 1999

Checkland, Peter, (1981) *Systems Thinking, Systems Practice*, John Wiley and Sons, 1981

Cilliers, Paul (1998) *Complexity and Postmodernism: Understanding Complex Systems*, Routledge, 1998

Cole, Michael (1998) *Cultural Psychology: A Once and Future Discipline*, Harvard University Press, 1998

D'Andrade, Roy, (1996)*The Development of Cognitive Anthropology*, Cambridge University Press, 1996

Engestrom, Y. and Middleton, David (1996)*Cognition and Communication at Work*, Cambridge University Press, 1996

Firesmith, Donald and Eykholt, Edward (1995) *Dictionary of Object Technology*, SIG Books, New York, 1995

Glascow J., Narayanan N., and Chandrasekaran B (1995) *Diagrammatic Reasoning: Cognitive and Computational Perspectives* AAAI Press/ The MIT Press, 1995

Hansen, Nohria and Teirney (HBR, March-April, 1999, pp. 106-118)

Havens, Charnell and Knapp, Ellen (1998) *Content, Community, and Computing* in Journal of Strategic Management, 1998

Hutchins, Edwin (1996) *Cognition in the Wild*, MIT Press, 1996

Inmon, W., Zachman, J. and Gelger, J. (1997) *Data Stores, Warehousing and the Zachman Framework: Managing Enterprise Knowledge*, McGraw-Hill, 1997

Kress, Gunther and van Leeuwen, Theo, (1996) *Reading Images: The Grammar of Visual Design*, Routledge, 1996

Jacobson, Ivar (1992) *Object Oriented Software Engineering: A Use Case Driven Approach*, Addison-Wesley, 1992

Jacobson, Ivar, Ericsson, Maria and Jacobson, Agneta (1994) *The Object Advantage: Business Process Reengineering with Object Technology*, Addison-Wesley, 1994

Larkin, Jill and Simon, Herbert, (1995) *"Why a Diagram Is (Sometimes) Worth Ten Thousand Words"* in Glascow J., Narayanan N., and Chandrasekaran B *Diagrammatic Reasoning: Cognitive and Computational Perspectives* AAAI Press/ The MIT Press, 1995

Mainzer, Klaus, (1994) *Thinking in Complexity: The Complex Dynamics of Matter, Mind and Mankind*, Springer-Verlag, London, 1994

Morecroft, John and Sterman, John (eds.), (1994) *Modeling for Learning Organizations*, Productivity Press, Portland, 1994

Odell, James (1995) *Object-Oriented Methods: Pragmatic Considerations*, Prentice-Hall, 1995

Odell, James (1998) *Advanced Object-Oriented Analysis and Design using UML*, Cambridge 1998

Odell, James and Martin, James, (1998) *Object-Oriented Methods: A Foundation: UML Edition*, Prentice-Hall, 1998.

Salomon, Gavriel (1996) *Distributed Cognitions: Psychological and Educational Considerations*, Cambridge University Press, 1996

Nardi, Bonnie (ed.)(1996) *Context and Consciousness: Activity Theory and Human-Computer Interaction*, MIT Press, Cambridge, 1996

Wertsch, J.V. (1991) *Voices of the Mind: A Sociocultural Approach to Mediated Action*, Cambridge, Harvard University Press, 1991.