# Next Generation Enterprise Modelling
## *The Role of Organizational Theory and Multi-Agent Systems*

Balbir Barn[1], Tony Clark[1] and Vinay Kulkarni[2]
[1]*Department of Computer Science, Middlesex University, The Burroughs, London, U.K.*
[2]*TRDDC, Tata Consulting Services, India*

Keywords: Organization Theory, Multi-Agent Systems, Actor Theory, Enterprise Modeling.

Abstract: This position paper proposes that the current enterprise modeling approaches are overly reliant on the know how or tacit knowledge of enterprise architects for addressing organizational challenges such as business-IT alignment. Furthermore, current modeling languages only encourage linear thinking. By drawing upon existing research on (computational) organization theory and multi-agent systems, we propose implementation requirements for a next generation enterprise modeling language that supports agent based simulation.

SCIENCE AND TECHNOLOGY PUBLICATIONS

## 1 INTRODUCTION

The modern enterprise is faced with the tricky challenge of responding to external drivers such as merger and acquisitions, or potential new markets, by adapting and managing internal change particularly with respect to business-IT alignment. Up to now, such a response has been dependent upon human expertise based on tacit knowledge and experience or "know how". Such a position is not sustainable with the rapid pace of change attributed to technology and globalization. This is confirmed with research that indicates that Strategic business-IT alignment has remained an ongoing concern for organizations (Luftman, 2004) and researchers have addressed the importance of alignment and in particular the need for congruence between business strategy and IT strategy (Chan and Reich, 2007).

One specific approach that has been used to bear upon the problem of business alignment is the role of Enterprise Architecture (EA) (Lankhorst, 2005). However, the predominant theme has focused on developing enterprise models that are descriptive in nature and hence needing human expertise for their interpretation (see ((Veken, 2013), (Zachman, 1987)) for two obvious examples). As a result, current approaches to Enterprise Modeling (EM) exhibit a high degree of latency in meeting key objectives such as alignment, adaptation *etc*. Thus EA in its current state does not readily lend itself to supporting the type of analysis that key decision makers typically utilize. Such stakeholders demand: ease of comprehension of the entire business so that decision-making can lead to efficient and effective change. In particular they require the ability play out various what-if (*i.e.*, what will be the consequences of such and such action) and if-what (*i.e.*, what would have led to such and such situation) scenarios to arrive at the right response, establish feasibility of the response, and estimate a ROI of the response. Thus ways of simulating an enterprise are needed and currently EA modeling approaches do not readily support this requirement.

Away from the EA modeling community, organizational theory and in particular computational organizational theory manifested in technologies such as multi-agent systems provides an opportunity to repurpose existing research outcomes to address the EA simulation and alignment conundrum. In doing so, this position paper proposes that next generation Enterprise Modeling languages should draw upon concepts from organizational theory and multi-agent systems in order that appropriate machinery can be implemented to support the simulation requirement.

The remainder of this paper is structured as follows: Section 2 presents key concepts from organizational theory and multi-agent systems. Specifically it draws upon the established research relationship that exists between the two areas. Section 3 presents the main contribution of the paper and proposes a novel language based approach to enterprise modeling for simulation. The proposal draws upon components, goal modeling and agent technologies. Section 4 discusses our plans for addressing the research challenges raised by the approach.

Barn B., Clark T. and Kulkarni V..
Next Generation Enterprise Modelling - The Role of Organizational Theory and Multi-Agent Systems.
DOI: 10.5220/0005097404820487
In *Proceedings of the 9th International Conference on Software Engineering and Applications* (ICSOFT-EA-2014), pages 482-487
ISBN: 978-989-758-036-9
Copyright © 2014 SCITEPRESS (Science and Technology Publications, Lda.)

## 2 ORGANIZATIONAL THEORY AND MULTI-AGENT SYSTEMS

Our starting premise is the view that there is a pressing need for next generation EM languages to address the requirements described in section 1 which can be broadly summarized as languages that: are machine manipulatable, support simulation through executability, and are model based. Such requirements raise two questions: What needs to be simulated? Secondly, what technologies can we deploy to support execution? In answering the first question we need to revert to Organizational Theory to understand the meaning of Organization and its constituting elements. Human organizations in particular have been the subject of detailed analysis from a range of disciplines including engineering, economics, psychology and sociology (Scott, 1992). The resulting analysis of the literature for organizational theory leads to a persuasive argument for the use of agent technology, particularly multi-agent systems as candidate technology for supporting the simulation/executability requirements identified.

Ours is an organizational society such that organizations are the dominant characteristic of modern societies. One rationale for the existence of organizations posited by Carley and Gassser is that they exist to overcome the cognitive, physical, temporal and institutional limitations of individual agency (Carley and Gasser, 1999). While there are many ways in which these limitations can be overcome and the structure, form or architecture of an organization contributes to such efforts, decades of research indicate that there is no optimal organizational design. Instead, the challenge morphs into one of adaptability and response to change. First we present here a necessarily brief overview of some of the key definitions and perspectives on organizations that underpin how we intend to articulate the concept of an organization in the context of the model driven enterprise. We first begin with a definition of the term organization recognizing that there are multiple definitions depending upon the perspective taken. The definition is reported from (Parsons and Jones, 1960):

> *Organizations are social units (or human groupings) deliberately constructed and reconstructed to seek specific goals.*

We explore this definition further by considering how the study of organizations has generally investigated the constituent elements of an organization and three dominant theoretical perspectives informing research. Leavitt identifies some core features of organizations (Leavitt, 1964):

**Social:** Structure regularized aspects of relationships among participants in an organization that may be both normative (embodying what ought to be) or factual order (actual structures).

**Participants:** individuals who in return for a variety of inducements make contributions to the organization. Participants may belong to more than one organization.

**Goals:** An organizational goal is a desired state of affairs which the organization attempts to realize. Goals are central to how an organization functions and are often vague or very specific.

**Technology:** This is the means by which work is performed in an organization. Technology can be interpreted as a manufacturing plant, the software systems enabling workers to perform work or even technical knowledge and skills of participants.

**Environment:** Organizations exist in a specific physical, socio-technical and cultural environment to which they must respond and adapt. All aspects of an organization are influenced and contextualized by the environment. For example, software systems are purchased from external providers or developed by technicians trained in some other organization.

These features are generic to organizations and can form the basis for extracting key concepts of an organization. Carley (Carley and Gasser, 1999) presents a similar set. Note that these features may vary in some way when viewed through a particular perspective or metaphor. The last century has seen three dominant perspectives (and overlaps) dominating research in organization theory: Organizations as Rational Systems; Organizations as Natural Systems and Organizations as Open Systems. A rational system perspective denotes a focus on efficiency and optimization and ultimately presents a reductionist model. The open systems perspective is of most relevance to us as it ranges from a simple clockwork view (a dynamic system with predetermined motions), cybernetic view (a system capable of self-regulation in terms of externally prescribed criterion such as a thermostat) to an open system (a system capable of self-maintenance based on throughputs of resources such as a living cell) (Buckley, 1967).

These views are categorized by Gazendam (Gazendam et al., 1998) and conform to essentially two categories: Classical Organizational theories and Systems theories. He suggests that classical theories have a strong correspondence to a machine metaphor where the organization as a whole consists of agents performing tasks in fixed structures consisting of agent tasks, communication paths and spatio-temporal orderings. Systems theories on the other hand view the organization comprising of sub-organizations fulfilling a specific function. Gazendam furthermore notes that: "System theories of organization are relatively poor because they only pay attention to the system level, and remain rather abstract."

Theories based on the machine metaphor have formed the basis of research on (multi-) agent-based system in the late 1980s and 1990s to study alternative viewpoints for describing organizations (Wooldridge, 2009). Here an *agent* is an autonomous and intelligent being such as a human or a simulator of a human realized by software (a computer agent) (Gazendam et al., 1998). Systems that are comprised entirely of computer agents have been used as simulations of organizations and correspondingly offer interesting perspectives on the study of organizations. Multi-agent systems (MAS) and the associated Computational Organizational Theory (Carley and Gasser, 1999) provide the collective apparatus for investigation.

Computational Organizational Theory (COT) aims to understand and model both human organizations and artificial organizations (multi-agent systems) that exhibit collective organizational properties such as the need to act collaboratively. Typical outputs of such research are the generation of new concepts and theories about organizing and organizations. Historically many applications and models have been constructed but our review of the current enterprise modeling literature indicates that COT has not yet been applied to some of the tricky problems of enterprise modeling such as Business-IT alignment discussed in the introduction.

There are immediate information processing requirements that are deducible from the definition of organization such as: information ubiquity, tasks, uncertainty distribution of organizational intelligence and necessity of communication through a model-based perspective. COT also suggests that: organizations are modelable, and so are manipulatable; are able to be designed to fit specific needs and there is an assumption that the costs of modeling and researching organizations in simulation mode rather than in vivo are lower (Carley and Gasser, 1999).

Key characteristics of organizations such as that described by Leavitt and Carley emphasize structure and behavior. (Hoogendoorn et al., 2007) propose these two aspects as necessary pre-requisites for modeling change when using MAS. In their proposal, organizations are described solely by the way groups and roles are arranged to form a whole. Related to this, Giorgini et al. use the i* framework (Yu, 1997) to define a series of architectural organizational styles which act as metaclasses and offer a set of design parameters for coordinating goals, actions and behavior and therefore govern how an organization functions (Kolp et al., 2006; Argente et al., 2006). Our position contributes to enterprise modeling technologies by drawing upon research outputs from COT to meet the needs of an adaptive organization located in a systematic understanding of socio-technical nature of an organization (Bean, 2010).

If MAS and COT are an appropriate way forward, then there are additional requirements for methods that can support COT based approaches. Those tasked with modeling enterprises need guidance that: "allow the description of social structures, permit the use of tools to perform project management, and include IDE or CASE tools that facilitate the analysis and design of MASs (Luck et al., 2005)". Furthermore, all methodologies need to contain enough abstractions to model and support MASs, which are usually structured as societies of agents that play roles and exchange information following predefined protocols (Isern et al., 2011). Isern et al. then go onto review a range of agent-oriented methodologies by evaluating their underlying meta models. Analysis of these meta models guides us toward the essential features of the language proposed in section 3.

We have posited that current approaches to EM presents a linear form of enquiry requiring tacit knowledge based on an Architect's know how that prevents scaling up to rapidly address "what if" type of questions. Adopting technologies based on MAS requires robust models for representing the complexity and dynamic nature of organizations as they respond to external business drivers. In particular then MAS can be used to provide simulation models for exploration of complex environments. Simulation models can be explanatory models that can help identify kinds of behaviour expected under specific conditions or they can predictive models that determine more precisely the the kind of behaviour a system will display in the future (Siebers and Aickelin, 2008).

Luck *et al.* propose a grouping of the agent-technologies, tools and techniques that can address these types of simulation for EM for theory building about an enterprise at three levels: Organizational-level (focusing on larger aggregations of structures; Interaction-level (collaboration, communication and decision making between agents) and Agent-level (learning and reasoning (Luck et al., 2005). Cross cutting concerns such as agent programming languages and methodologies (noted earlier) provide practical steps towards realization of agent systems.

In the next section, we discuss this partitioning has been used to influence our proposal.

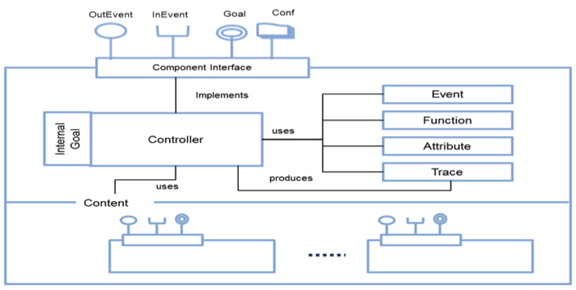

Figure 1: Component Abstraction (Core Concepts).

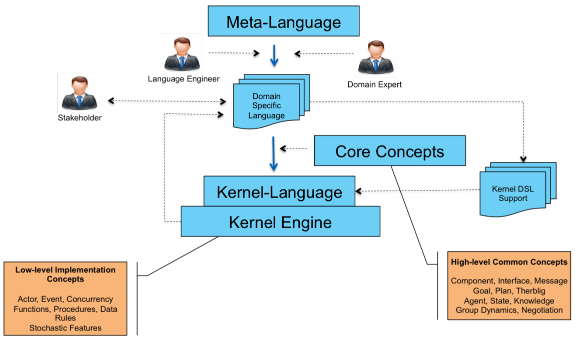

Figure 2: The EASE-Y Architecture.

## 3 NEXT GENERATION EM

We posit that any approach that is derived from ideas from the previous sections relies on being able to represent and process an organization that is expressed in terms of a component-based abstraction. We envisage a product-line approach (Reinhartz-Berger et al., 2013) whereby a suite of tools based on this abstraction is used to facilitate a collection of different organization analysis and simulation activities. Each activity will constitute a domain, *e.g.*, cost analysis, resource analysis, mergers and acquisition, regulatory compliance. In principle, each new domain will require a new domain specific language to represent the concepts. How should such a proliferation of domains be accommodated by a single component abstraction?

Our proposal is to construct an extensible kernel language that is used as the target of translations from a range of domain specific languages (DSLs). Each DSL supports an organization analysis and simulation use-case. We then aim to construct a virtual machine for the kernel language so that it is executable. Model execution supports organization simulation and some analysis use-cases. Links to external packages such as model-checkers will complete the analysis use-cases.

The use of a single kernel language provides a single focus of development effort and can help minimize the problem of point-to-point integration of analysis methods. Our proposal is that a small core collection of concepts, including *component*, *interface*, *goal*, *event*, *function* as shown in figure 1, are a suitable basis for most types of analysis and simulation use-case and therefore the kernel language will be defined in terms of these concepts.

Given its ability to accommodate multiple simulation and analysis use-cases, we envisage the language being the basis of a suite of organizational modeling, simulation and analysis tools, presented in the form of a single integrated extensible meta-tool EA Simulation Environment (*EASE-Y*) shown in figure 2. Since organizational information is likely to be very large (at least many tens of thousands of model elements) it is important the tool is implemented efficiently, is scalable, supports distributed concurrent development and is flexible in terms of its architecture. To this end we aim that the kernel language should be compiled to a machine language running on a dedicated kernel engine, the language integrates with standard repository technology, and can run equally well on single machines, networked machines and via the cloud.

Organizations consist of many autonomous components. Components are organized into dynamically changing hierarchical groups, operate concurrently, and manage goals that affect their behavior. We aim for the kernel language to reflect these features by having an operational semantics based on the Actor Model of Computation (AMC) (Hewitt, 2010) and its relation to organizations, or iOrgs (Hewitt, 2009). Actors have an address and manage an internal state that is private and cannot be shared with other actors in the system.

Execution proceeds by sending asynchronous messages from a source actor to the address of a target actor. Synchronous messages can be achieved by sending an actor in an asynchronous message to which the result should be sent. Each message is handled in a separate execution thread associated with the target of the message and the message itself (collectively referred to as a task). During task-execution an actor may choose to change its state and behavior (becoming a new actor) that is immediately available to process the next message sent to the target address.

Our claim is that the AMC provides a suitable basis for execution and analysis of the concepts discussed in section 2. Actors, sometimes individually and sometimes collectively, can be used to represent the features of a component. The rest of this section lists the key features that must be supported by the kernel language and how the actor approach can support them:

[**adaptability**] This is required because organizational components may change dynamically during a simulation. Resources, individuals, and even departments may move location, and have an affect on results. Furthermore, the behavior of a component may change over time as information changes within the system. An actor can, in principle, change behavior as a result of handling each message.

[**modularity**] Each part of an organization is intended to perform a business function that can be expressed in terms of a collection of operations. The internal organization in terms of people, IT systems and the implementation of various business processes is usually hidden. The AMC provides an interface of message handlers for each actor. Both the state and the implementation of the message interface are hidden from the outside. The specification of an actor in terms of its external interface can be expressed in terms of LTL formulas that constitute the external goal for a component.

[**autonomy**] A key feature of an organization is that the behavior of each sub-component is autonomous. A particular department is responsible for its own behavior and can generate output without the need for a stimulus. The AMC is highly concurrent with each actor being able to spawn multiple threads and over which other actors have no control (unless granted by the thread originator).

[**distribution**] An organization may be distributed and this may be an important feature of its simulation. Furthermore, we have a requirement that the tooling for organizational analysis and simulation should support distributed concurrent development. The AMC associates actors with addresses to which messages are sent. Execution does not rely on the particular location of the actor (i.e. the mapping between the address and the actor behavior) that can be in the same address space, via a network connection or in the cloud.

[**intent**] In addition to autonomous behavior, an organization component exhibits intent. This might take the form of an internal goal that guides the behavior of the component to ensure that it contributes to the overall mission of the organization. Although actors do not directly provide support for such goals, we intend to use results from the field of Multi-Agent Systems (Wooldridge, 2009) where support for goal-based reasoning is provided within each agent when determining how to handle messages.

[**composition**] An organization is an assembly of components. As noted above, the topology of an organization may be static or dynamic. Actors can be nested in more than one way. Actor behaviors are declared and new actors are dynamically created with an initial behavior (much like Java classes). The scope of actor behaviors can be nested to provide modularity. Adding a dynamically created actor to the state of a parent actor provides composition. Such actors can be sent as part of messages. If the source actor retains the address, then the communicated actor becomes shared between the source and the target of the message.

[**extensibility**] Our aim is to support a number of simulation and analysis use-cases. As such the kernel language will need to support a collection of independent domains. Whilst we expect the DSLs to target the kernel language it is likely that each domain will have its own fundamental concepts and actions (so-called Therbligs, (Stanton, 2006)). We envisage such domain-specific features being defined in the kernel language and then pre-loaded to form an augmented target language for DSL translations.

[**event-driven**] Organizational components cannot rely on when communications occur and where they originate. In addition, a component may simply cause an event to occur without knowing who will consume the event. This is to be contrasted with message-based communication where the target is always known to the source and where sometimes the message carries information about the source that becomes available to the target. The AMC is based on message passing where the source knows the address of the target. Given that the kernel language is the target of DSL transformations, support for event-based communication becomes an architectural issue where events are simply messages that are sent to an actor container that is responsible for delivering event-messages to dynamically changing collections of actors. Providing that the transformation establishes the correct assembly of actors and conforms to an appropriate message passing protocol then component events are supported without needing to make them an intrinsic part of the kernel.

## 4 CONCLUDING REMARKS

This position paper has proposed that current generation enterprise modeling languages and technologies support a linear form of enquiry that requires tacit knowledge based on an Architect's know how. Such an approach prevents scaling up to rapidly address "what if" type of questions that face organizations as they seek to adapt to respond to ongoing change. At an abstract level, these types of requirements have been studied in other disciplines, including economics, political science, philosophy and linguistics leading to computation based organizational theories and technologies for describing agent interaction, communication and decision-making. We have presented an argument that traces a route through (computational) organization theory to propose that next generation enterprise modeling languages should address COT and multi-agent system approaches to provide a rich simulation platform that supports both explanatory models and predictive models for the "what if" question. In doing so, we recognize that there are open-ended research questions around methodology and proposed the simulation platform. We plan to validate our proposition in a number of ways. We are currently developing a collection of representative case studies based on real-world data in a laboratory setting. One case study illustrates how the proposed ideas and techniques can help data-driven decision making in an IT services providing organization. Another case study will address merger and acquisition problem in wealth management domain. We intend to run co-design workshops with Business Management domain experts in order to evaluate their response to our proposals. We are currently extending $\mu$LEAP (Clark and Barn, 2014) to be the target kernel language. We plan to design and implement the kernel language meta-model as a virtual machine, possibly using multiple Java VMs as targets. Scalability of the platform in terms of the number of runtime virtual machines is a critical factor for acceptability of the platform in practice. Recent research by Bolz and Tratt (Bolz and Tratt, 2013) seems a promising direction in this regard.

## REFERENCES

Argente, E., Julian, V., and Botti, V. (2006). Multi-agent system development based on organizations. *Electronic Notes in Theoretical Computer Science*, 150:55–71.

Bean, S. (2010). Re-thinking enterprise architecture using systems and complexity approaches. *Journal of Enterprise Architecture*, 6(4):7–13.

Bolz, C. F. and Tratt, L. (2013). The impact of meta-tracing on vm design and implementation. *Science of Computer Programming*.

Buckley, W. (1967). *Sociology and modern systems theory.* Prentice-Hall.

Carley, K. M. and Gasser, L. (1999). Computational organization theory. *Multiagent systems: A modern approach to distributed artificial intelligence*, pages 299–330.

Chan, Y. E. and Reich, B. H. (2007). It alignment: what have we learned? *Journal of Information technology*, 22(4):297–315.

Clark, T. and Barn, B. S. (2014). Outsourcing service provision through step-wise transformation. In *Proceedings of the 7th India Software Engineering Conference*. ACM.

Gazendam, H. W., Jorna, R. J., et al. (1998). *Theories about architecture and performance of multi-agent systems*. University of Groningen.

Hewitt, C. (2009). Norms and commitment for iorgs (tm) information systems: Direct logic (tm) and participatory grounding checking. *arXiv preprint arXiv:0906.2756*.

Hewitt, C. (2010). Actor model of computation: scalable robust information systems. *arXiv preprint arXiv:1008.1459*.

Hoogendoorn, M., Jonker, C. M., Schut, M. C., and Treur, J. (2007). Modeling centralized organization of organizational change. *Computational and Mathematical Organization Theory*, 13(2):147–184.

Isern, D., Sánchez, D., and Moreno, A. (2011). Organizational structures supported by agent-oriented methodologies. *Journal of Systems and Software*, 84(2):169–184.

Kolp, M., Giorgini, P., and Mylopoulos, J. (2006). Multi-agent architectures as organizational structures. *Autonomous Agents and Multi-Agent Systems*, 13(1):3–25.

Leavitt, H. J. (1964). Applied organization change in industry: structural, technical and human approaches. *New Perspectives in Organizational Research*, 55:71.

Luck, M., McBurney, P., Shehory, O., and Willmott, S. (2005). Agent technology: computing as interaction (a roadmap for agent based computing).

Luftman, J. (2004). Assessing business-it alignment maturity. *Strategies for information technology governance*, 4:99.

Parsons, T. and Jones, I. (1960). *Structure and process in modern societies*, volume 3. Free Press New York.

Reinhartz-Berger, I., Cohen, S., Bettin, J., Clark, T., and Sturm, A. (2013). *Domain Engineering*. Springer.

Scott, W. R. (1992). *Organizations*. Prentice-Hall Englewood Cliffs, NJ.

Siebers, P.-O. and Aickelin, U. (2008). Introduction to multi-agent simulation. *arXiv preprint arXiv:0803.3905*.

Stanton, N. A. (2006). Hierarchical task analysis: Developments, applications, and extensions. *Applied ergonomics*, 37(1):55–79.

Veken, K. V. d. (2013). Enterprise architecture modelling to support collaboration-the archimate language as a tool for communication.

Wooldridge, M. (2009). *An introduction to multiagent systems*. John Wiley & Sons.

Yu, E. (1997). Towards modelling and reasoning support for early-phase requirements engineering. In *Requirements Engineering, 1997., Proceedings of the Third IEEE International Symposium on*, pages 226–235. IEEE.

Zachman, J. A. (1987). A framework for information systems architecture. *IBM systems journal*, 26(3):276–292.