SEMANTICS AND PRAGMATICS IN ENTERPRISE ARCHITECTURE THROUGH TRANSACTION AGENT MODELLING

Ivan Launders, Simon Polovina
Conceptual Structures Research Group, Sheffield Hallam University, Arundel Street, Sheffield, U.K.

Richard Hill
Distributed and Intelligent Systems Research Group, University of Derby, Derby, U.K.

Keywords: Semantics, pragmatics, Enterprise architecture, Transaction agent modelling.

Abstract: Enterprise architectures comprise of complex transactional information systems that perform repetitive and bespoke business transactions to meet business goals. Frameworks for enterprise architectures have been widely adopted to organise design thinking about the architectural components as well as to provide a description of architecture artefacts. We note various shortcomings of these framework approaches, giving rise to how semantics and pragmatics should evolve in enterprise architectures through Transaction Agent Modelling (TrAM). We accordingly outline steps for capturing and modelling the semantics in business transactions for enterprise architecture.

1 INTRODUCTION

Frameworks for enterprise architectures have been widely adopted to help organise design thinking about complex transactional information systems (TOGAF 2009, and Zachman 1987). Sowa & Zachman provided an original vision of what a contemporary enterprise architecture should be (Sowa and Zachman 1992), extending the vision of an Information System Architecture (ISA) framework to show how it could be formalised in the notation of Conceptual Graphs (Sowa 1984). Zachman recognised that modelling tools and techniques of the day (e.g. entity-relationship diagrams, object-oriented systems, UML) were specialised for different purposes and that by concentrating on one aspect, a technique can lose sight of the overall information system and how it relates to the enterprise (Fowler 2004, Jacobson et al. 1992, Sowa and Zachman 1992). Sowa and Zachman's approach to Information System Architecture (ISA) was to use a framework to provide a system architecture scope; an enterprise or business model; a system model, and finally a technology model and its subsequent model components (Sowa and Zachman 1992). Today an ISA would be referred to as an Enterprise System Framework.

Architecture development frameworks provide a structure within which the key components of the architecture and the relationship between those components are defined (Sowa and Zachman 1992, TOGAF 2009, Zachman 1987). A framework helps to organise our thinking about software architecture and provides a description of artefacts. It helps ensure that the semantics in an enterprise are widely understood. Attributes of a good architecture development framework include:

- consistency and structure;
- capturing the 'kite' or high level process;
- incorporating a variety of constructs at different levels of abstraction;
- a defined, enabling process for developing the architecture;
- a description of the artefacts that will be produced during the work of the architecture development;
- a clearly described process.

The Open Group Architecture Framework (TOGAF) provides a significant move towards

Launders I., Hill R. and Polovina S.
SEMANTICS AND PRAGMATICS IN ENTERPRISE ARCHITECTURE THROUGH TRANSACTION AGENT MODELLING.
DOI: 10.5220/0003268302850291
In *Proceedings of the Twelfth International Conference on Informatics and Semiotics in Organisations (ICISO 2010)*, page
ISBN: 978-989-8425-26-3

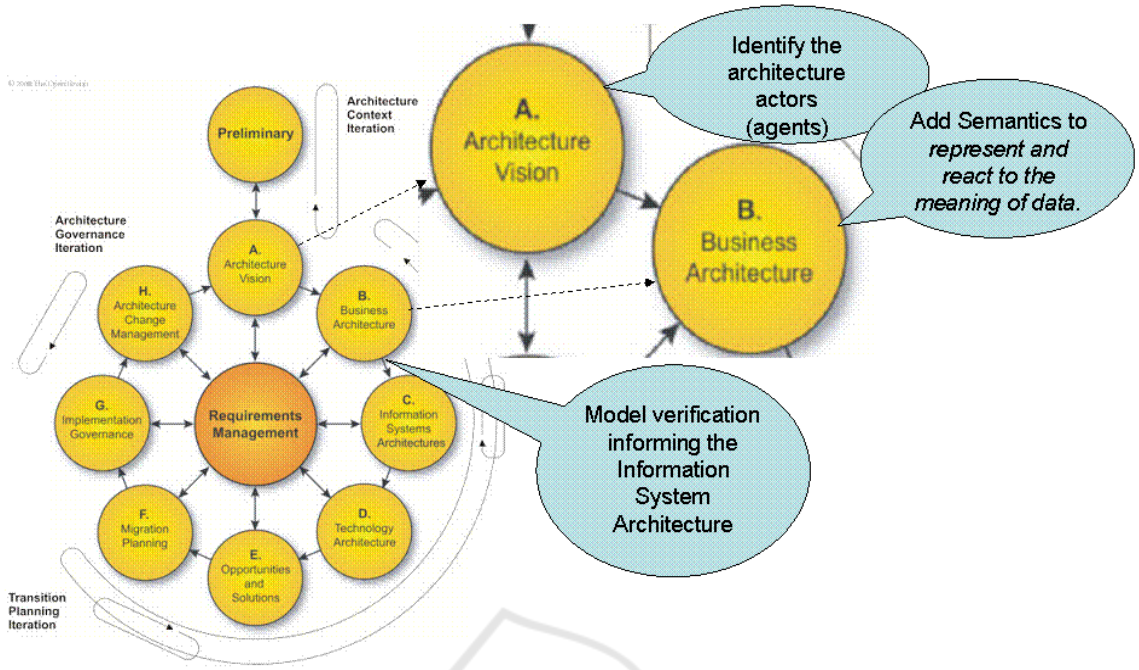

Figure 1: TOGAF ADM with added semantics

working with a formal Enterprise Architecture framework, by the development of a meta-Framework (TOGAF 2009) that is designed to be customised and could be extended to add semantics. TOGAF was first developed in 1995 based on the US Department of Defence Technical Architecture. It provides a tool for assisting the acceptance, production, use and maintenance of architectures. The approach is an iterative process model focusing on architecture types including:

- *Business Architecture*: The business strategy, governance, organisation and the key business processes. Semantics would enhance this architecture type.
- *Data Architecture*: The structure of an organisation's logical and physical data assets and data management resources.
- *Application Architecture*: The blueprint for applications, and their relationships to the core business processes of the organisation.

The *Data Architecture* and *Application Architecture* comprise the *Information Systems Architecture* of TOGAF.

Capturing a business process "As Is" provides a record of the business as it currently works; in TOGAF this is referred to as a "baseline". Analysing enterprise business processes with high level business goals, such as improving efficiencies and performance, and then having the ability to be able to model, automate, and visualise those potential improvements, offers business transactions refinement at a semantic level. Transaction Agent Modelling (TrAM) extends requirements capture using the rigour of Conceptual Graphs (CG) and Resource-Events-Accounting, latterly referred to as the Resource-Event-Agent (REA) model (Geerts and McCarthy 1991, McCarthy 1982, McCarthy 1979, and Polovina 1993).

We outline an automated approach to assist designers with the capture of semantics in enterprise transactions through the use of a generic Transaction Model (TM), allowing for business transactions to be enriched and refined at the early stage in the design process. We explain how an improved framework that places greater emphasis upon the capture of semantics in business transactions through the automation of CG can assist an enterprise architect (Hill and Polovina 2008, Polovina 2007, Polovina and Hill 2009, and Launders et al. 2009).

2 SEMANTICS AND PRAGMATICS IN BUSINESS ARCHITECTURE

The linguistic understanding of the word "semantics" is the study of meaning itself (Sowa 1984). Pragmatics studies how the basic meaning is

related to the current context and the listener's expectations. Syntax studies the grammar rules for expressing meaning in a string of words (Sowa 1984, Stamper 1996). Semantics determines the literal meaning. Other factors, which relate language to the world, are called pragmatics. In the context of achieving successful communications in an enterprise, stake-holders in business transactions need to be consistent in the use of language. We focus attention on where the semantics are in business transactions and how we capture and work with them in enterprise architecture. The Semantics of Business Vocabulary and Business Rules (SBVR, http://www.businessrulesgroup.org/sbvr.shtml) group provides an approach that enables people and organisations to treat business, legal, and educational knowledge in a productive and consistent way. SBVR aims to combine the valuable aspects of logic, natural language, business rules, and conceptual modelling. Business knowledge can be described using:

- a business concepts catalogue;
- an association of business fact types;
- a business rules catalogue.

A business concepts catalogue is identified by SBVR, as a major step forward, in that consistency in the use of concepts in business rules is important in ensuring the quality of business rules. A description should be consistent with Sowa's definition, that a conceptual catalogue shows how the form can be applied to the words and concepts (Sowa 1984).

One of the areas of future development in architectural frameworks such as TOGAF could be in the capture of semantics in the business architecture. Figure 1 provides the Architectural Development Method (ADM) for TOGAF illustrating where to add semantics into the ADM phases. The Business Architecture phase 'B' could be extended to capture the semantics in high level business transactions.

A TOGAF Architecture Vision as shown in Figure 1 starts by articulating business requirements implied in new business functionality to meet business goals. It then implies a technical architecture requirement. Two key elements of this step include identifying:

- *Human Actors:* Identify human actors and their place in the business model, the human participants and their roles.
- *Computer Actors:* Identify computer actors and their place in the technology model, the computing elements and their roles.

TOGAF states that a variety of modelling tools and techniques may be employed, if deemed appropriate, therefore we have considered a Transaction Agent Modelling framework referred to as TrAM (Hill 2005). TrAM is a requirements elicitation framework for agent-oriented software. Activity Models (also called Business Process Models) describe the information exchange functions (internal and external) associated with the enterprise's business activities, from a data perspective. Activity Models are hierarchical in nature, capturing the activities performed in a business process, and TOGAF refers to these as ICOMs (inputs, controls, outputs, and mechanisms/ resources used) of those activities. Models represented as use case diagrams can describe either business processes or systems functions, depending upon the focus of the modelling effort. In TrAM we refer to Transactional Use Cases (TUC) which are specifically focused on enterprise transactions. TUC describe at a high level the primary business transactions in an enterprise in terms of use cases and internal and external actors, corresponding to business transactions. TrAM provides a greater level of focus, capturing semantics within the business transaction initially captured in TUC, thus developing upon TOGAF's more general approach to activity and use case modelling.

An experienced software designer would consider enterprise architecture from a human and social level but they would tend to deal with it intuitively and informally using experience and domain knowledge from previous work (Stamper 2007). CG allow the designer to represent knowledge and to use interactive tools such as Amine (http://amine-platform.sourceforge.net/) for representing natural language in a structured, knowledge representation language. Stamper's "Semiotic Ladder" helps identify that information systems function adequately when signs are handled correctly on every architectural level, and that ineffective systems tend to ignore problems on one or more levels, typically the upper three levels of the Semiotic Ladder (social world, pragmatics and semantics) (Stamper 1996, Stamper 2007). Technical architecture levels are often more accurately specified and designed, whereas this is more unlikely in the analysis and design of semantics and pragmatics in enterprise architecture.

Conceptual analysis enables software designers to define the schemata for enterprise architecture. Applying conceptual analysis through TrAM provides the focus upon an enterprise transaction through the rigour of CG by providing model

checking to assist in the early requirements capture (Hill et al. 2005). The results of the analysis and the description of the concepts within a particular domain are known as an *ontology* (Gruber 1993). An ontology contains the understanding of knowledge in a given domain and creating that ontology requires input from the human information functions (semantics, pragmatics, and social world) through domain knowledge. Creating that ontology containing enterprise semantics can significantly benefit from the CG approach particularly in conjunction with software tools for the development of intelligent systems. This process is likely to be iterative.

3 SEMANTICS AND TRANSACTION AGENT MODELLING

The main intent of semantics in communications is to give machines better access to information so they can be information intermediaries in support of humans (Burners-Lee et al. 2001). When agents communicate with each other, there needs to be a means of exchanging meaning through a transaction, so agent A has the same conceptual understanding as agent B in a business transaction. If we accept that agents may not use the same terms to mean the same things, we need a way to discover what another agent means when it transacts (Uschold 2001). In order for transactions to happen, every agent will need to declare or catalogue what terms it is using and what they mean, much the same as human agents do linguistically through a glossary. This specification is referred to as the agent's ontology (Gruber 1993). An ontology can describe meaning as a formal specification of the terms in a domain and the relationships between them, using a common vocabulary for agents who need to share information in that domain.

Figure 2: Software Agent Encoding.

Semantics must be accessible to software agents and therefore need encoding in some form of logic (Uschold 2001), as illustrated in the community care example in Figure 2. This will enable software agents to use automated reasoning to accurately determine the meaning of other software agents. In practice there are many difficulties to overcome for this to happen reliably and consistently. However before the stage of ontology representation a designer needs to capture and model the semantics in business transactions. Whilst enterprise architecture modelling can facilitate the visualisation of business processes, there are limitations, namely: the accuracy of the process model; the completeness of a model, and therefore being aware of what is missing; if there are any efficiencies to be had which an enterprise could benefit from.

In the context of semantics TrAM focuses on the machine processing semantics for REA type business transactions. The approach TrAM takes is to capture and represent these semantics, in order that a base ontology can be derived, but also to support the reasoning of new concepts as and when they occur. TrAM theory defines that enterprise system designs will include the following:

- *Model fundamentals*: Providing a complete Transactional Use Case diagram (TUC) capturing the transactional behaviour of the initial use case. A close mapping between TUC and CG, translating that transactional behaviour and adding semantics through comprehensive analysis with CG, including co-referent links and a supporting type hierarchy (Hill et al. 2005).
- *Model Visualisation*: Visualisation of business rules, using Peirce logic. CG are a system of logic based on Peirce's Existential Graphs (Roberts 1973). Proof of the enterprise business rules, specialisation and projection within the proof of the design rules.
- *Model Automation*: CG tools such as Amine (http://amine-platform.sourceforge.net/) provide good use of transferring the initial transaction model analysis into an Amine ontology with accurate type hierarchy and TM. Integration of the model with a conceptual catalogue (CC) showing how form can be applied to words and concepts in the transaction model. Use of the ontology to achieve a successful projection with the inclusion of the business rules in the ontology.

3.1 TrAM Model Automation

Model automation involves building the ontology for the TM as well as integrating a Conceptual Catalogue into the ontology to describe the form and meaning of the concepts in the model (Hill and Polovina 2008, Uschold 2001, Launders et al. 2009).

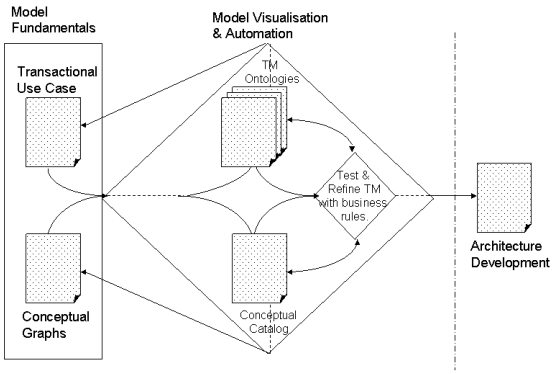

Figure 3: TrAM, with automation.

Figure 3 provides an illustration of TrAM with automation, where the designer transforms a paper based CG analysis (Model Fundamentals & Model Visualisation) into a software model for verification (Model Automation) through CG operations using Amine. The steps for automation expand upon Hill's original TrAM framework (Hill et al. 2005). These steps are as follows:

- *Model Fundamentals*
  1. Capture Transactional Use Case logic (TUC)
  2. Create a Transaction Model
     a) Transform TUC into CG
     b) Integrate CG with the generic TM
     c) Iterate TM with TUC
- *Model Visualisation*
  3. Visualise the TM using Peirce Logic to highlight any further requirements.
- *Model Automation*
  4. Create a Conceptual Catalogue for the transactional terms used in the TM
  5. Verify the semantics and logic captured in the TM (Inference against models and verify using Amine)
     a) Create and Refine the Type Hierarchy;
     b) Build the Ontology for the TM (or Ontologies if there are more than one)
     c) Integrate the Conceptual Catalogue
  6. Specialise the TM through business rules (Test and refine the TM with business rules using Amine).

The 'transactional use case' (TUC) diagram captures high level transaction logic, to be transferred into Transaction Model Conceptual Graphs, representing concepts and relations for each of the high level transactions identified. The TUC capture involves analysing key transactional facts from the case study narrative. Central to this analysis and analogous to Zachman is identifying the 'What' (Economic Resources?), 'How' (Economic Events?), 'Who' (agents?), and 'Why' (business goals).The subsequent step involves mapping those use cases into CG, translating and adding semantics through the use of a conceptual catalogue and Model Verification. The resulting Transaction Model (TM) ontology is then tested and refined specialising the transaction logic through the application of business rules. The resulting design artefacts include:

- a refined Transaction Use Case (TUC);
- a transformation of TUC into CG;
- a Transaction Model in CG consisting of a type hierarchy together with domain constraint rules modelled with Peirce Logic;
- Conceptual Catalogue of domain specific terms, automated and re-usable;
- an automated TM Ontology which has been tested and refined through CG operations using Amine.

These artefacts result in a design specification for the eventual enterprise architecture that does not impose a particular implementation and serves to complement architecture framework that lack a semantic requirements gathering stage such as TOGAF.

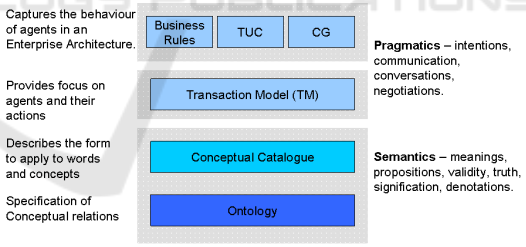

Figure 4: TrAM Semantics and Pragmatics.

Figure 4 illustrates the design artefacts in layers showing how they sit in relation to each other and where semantics and pragmatics are modelled. The ontology layer through a conceptual catalogue (CC) describes the form to apply to words and concepts, as canons and definitions in capturing the semantics (http://cg.huminf.aau.dk/Module_III/1152.html).

The transaction model (TM) layer following a generic TM uses the CC and ontology capturing the negotiation between events and resources. A top layer including the business rules specialises the enterprise transaction.

4 SIMULATION OF INDUSTRIAL PRACTICE

Our experience has been informed through an innovative approach to LTA (learning, Teaching and Assessment), where student design teams experience the application of this emerging computing theory in architecture for enterprise applications. We have used a combination of case studies both 'real world' and 'fictitious' as a simulation of industrial practice, for example a mobile NHS case study allowing clinicians to be able to access patient records during visits to patients is a direct capture of an industrial experience. Each case study provided business transactions with different business settings. One of the goals of analysing design data from multiple case studies (or cross case) is to examine if the events and process on one well described setting can occur in a different setting. Each case study contained a narrative account of a situation focusing on the business transactions and exchange of resources for events.

5 CONCLUDING REMARKS

Complex enterprise transactions need framework approaches which create deeper understanding of the semantics in a business domain. A semiotic perspective offers a route to deeper understanding, and an emphasis on the sign and communications character of information systems (Goldkuhl and Agerfalk 2002, Stamper 1996, Stamper 2007). Early requirements modelling helps examine the concepts and semantics in transactions and therefore visualise and deepen the understanding of how the business will actually perform under specific conditions. Visual models are important for clarifying the meaning in business transactions at different levels of abstraction. However, abstract models have their limitations such as how closely they actually relate to an enterprise and therefore how complete they are in terms of capturing semantics and visualising possible efficiencies. We propose therefore, that through our illustration of integrating the semantics (and semiotics) of TrAM, with the development of CG automation tools such as Amine, together with TOGAF, addresses the sufficient complexity of the real world in which businesses operate.

REFERENCES

Berners-Lee, T., Hendler, J., Lassila, O. (2001) "The Semantic Web", Scientific American, Vol. 284, Issue 5.

Fowler. M. (2004) UML Distilled (Third Edition), Addison-Wesley, 103-104.

Geerts, G. L., McCarthy. W. E. (1991). "Database Accounting Systems", in Information Technology Perspectives in Accounting: an Integrated Approach, Chapman and Hall, 159-183.

Goldkuhl, G. Agerfalk, P. J. (2002) "Actability: A Way To Understand Information Systems Pragmatics", In Liu, K et al. (eds.), Coordination and Communication Using Signs: Studies in Organisational Semiotics – 2, Kluwer Academic Publishers, Boston, 2002

Gruber, T.R. (1993). A Transaction Approach to Portable Ontology Specification. Knowledge Acquisition 5: p199-220

Hill, R., Polovina, S., (2008) "An Automated Conceptual Catalogue for the Enterprise", Supplementary Proceedings of 16th International Conference on Conceptual Structures (ICCS 09): Conceptual Structures: Knowledge Visualization and Reasoning, Toulouse, France, July 2008, Eklund, P., Haemmerlé, O. (Eds.), CEUR-WS, Vol-354, 99-106.

Hill, R., Polovina, S., Beer, M. D. (2005) "From concepts to agents: Towards a framework for multi-agent system modelling", Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS), Utrecht University, Netherlands, ACM Press, 1155-1156.

Jacobson, I., Christerson, M., Jonsson, P. and Overgaard, G. (1992) Object-Oriented Software Engineering. Wokingham, England: Addison-Wesley.

McCarthy, W. E., (1982) "The REA Accounting Model: A Generalized Framework for Accounting Systems in a Shared Data Environment", The Accounting Review, 554-578.

McCarthy, W. E., (1979) "An Entity-Relationship View of Accounting Models", The Accounting Review, 667-686.

Polovina, S., (2007). "An Introduction to Conceptual Graphs", Proceedings of the 15th International Conference on Conceptual Structures (ICCS 2007): Conceptual Structures: Knowledge Architectures for Smart Applications, July 2007, Sheffield, UK; Priss, Uta; Lecture Notes in Artificial Intelligence (LNAI 4604), Springer, 1-15.

Polovina S., Hill, R. (2009) "A Transactions Pattern for Structuring Unstructured Corporate Information in Enterprise Applications", International Journal of Intelligent Information Technologies, April-June 2009, Vol. 5, No. 2, IGI Publishing, 34-48.

Polovina, S. (1993) "Bridging Accounting and Business Strategic Planning Using Conceptual Graphs", Conceptual Structures: Theory and Implementation, Pfeiffer, Heather D; Nagle, T. (Eds.), LNAI, Springer-Verlag, Berlin, 312-321.

Sowa, J. F., (1984). Conceptual Structures: Information Processing in Mind and Machine, Addison-Wesley.

Sowa, J., Zachman, J.A. (1992) "Extending and formalizing the framework for information systems architectures", IBM systems journal VOL 31, No 3, 1992

Stamper, R. (1996). Signs, Norms, and Information Systems. In B. Holmqvist et al. (Eds.), *Signs at Work*. Berlin, Germany: Walter de Gruyter (pp. 349-397).

Stamper, R. Stumbling across a "soft Mathematics" while exploring some issues of Organisation, Law and Metaphysics. Unpublished, Proposed for ICCS 2007.

TOGAF, The Open Group Architecture Framework (TOGAF), Version 9, Enterprise Edition.

Launders, I. (2009). "Socio-Technical Systems and Knowledge Representation" Whitworth , B., & De Moor, A. (Eds.). (2009). Handbook of Research on Socio-Technical Design and Social Networking Systems. Hershey; 1,034 pp

Launders, I., Polovina, S., Hill, R., (2009). "Exploring the Transaction through Automating TrAM", ICCS 2009: Supplementary Proceedings of the 17th International Workshops on Conceptual Structures; , Moscow: Springer

Roberts D. D, (1973) The existential Graphs of Charles S. Peirce. The Hague, The Netherlands: Mouton.

Uschold, M., (2001). "Where are the Semantics in the Semantic Web". Autonomous Agents Conference, Montreal, June 2001.

Zachman, J.A. "A Framework for Information Systems Architecture," *IBM Systems Journal* 26, No. 3**,** 276-292 (1987).