SYNTAX-DIRECTED TRANSLATION SCHEMES FOR MULTI - AGENT SYSTEMS CONVERSATION MODELLING

Ana Fred*, **Joaquim Filipe****

**Instituto de Telecomunicações,*
Instituto Superior Técnico, Lisboa,
Av. Rovisco Pais, 1049-001 Lisboa, Portugal
e-mail: afred@lx.it.pt

***Escola Superior de Tecnologia de Setúbal,*
Campus do Instituto Politécnico de Setúbal,
Rua Vale de Chaves, Estefanilha, 2914 Setúbal, Portugal
e-mail: JFilipe@est.ips.pt

Abstract: In modern organisations the monolithic information systems of the past are being gradually replaced by networked systems, enabling distributed computing often based on multi-agent system architectures. This new paradigm enables the use of information systems support in new areas of organisational activity, especially those involving the interaction of business agents. All communication-intensive business processes based on formal conversations, i.e. partially ordered sets of communicative acts transmitted among a set of agents, qualify as candidates to, at least partial, automation. Still a very active area of research, this paradigm has been studied in areas such as distributed artificial intelligence, organisational simulation and workflow management. However, in all these areas the basic problem is the adequate representation of agent conversations. In this paper we present a formal method for conversation representation that is inspired in syntactic pattern recognition methods, specifically syntax-directed translation schemes. This method has a clear semantics that can be easily given a declarative implementation, thus becoming flexible enough to accommodate on-line extensions and exception handling.

Key words: Multi-Agent Systems, Syntax-directed Translation Schema, Attributed Grammars, Speech Act Theory, Distributed Artificial Intelligence.

1. INTRODUCTION

Given the approximation of the multi-agent system paradigm to the functioning of human organisations it is only natural that computing metaphors inspired in Organisation Theory and Speech Act Theory have been adopted to model the behaviour of information systems. Amongst these are the concepts of organisational role, conversation, intention and action.

Similarly to what is observed in human organisations, in addition to the many advantages of task decomposition and parallel execution, one of the problems of multi-agent systems is the coordination of the various agents. We adopt the assumption that there are two main coordination mechanisms: norms and communication. Behavioural norms define standard accepted types of behaviour, thus enabling large parts of the organisation to function in coordination without much communication, whereas in situations of uncertainty communication becomes the essential coordination mechanism. In this paper we focus mainly on the communication aspects, but in organisations even communication protocols themselves are subject to norms. The way agents communicate depends on the organisational roles they are playing in that particular conversation. In the first stages of conversation, each agent selects an adequate conversation plan (CP), accordingly to the intentions they have and/or expect the other agent to have. An agent may or may not know the CP of the other agent; however coordination will be improved if each agent has access to the other agent CP because that permits more precise expectations on the other's behaviour.

Many of the current multi-agent systems (Labrou, 1997; Barbuceanu et al., 1997; Singh, 1997) use speech acts, as originally proposed by (Austin, 1962) or communicative acts (Habermas, 1984), as the conceptual basis for representing conversations. A conversation is defined as a partially ordered set of communicative acts transmitted among a set of agents. Conversations are also frequently used in business processes modelling and, since the seminal work of Winograd and Flores (1986), they are often formally represented using various types of finite state machines. In this paper, the formalism for the representation of multi-agent conversations is viewed under an *interpretation* paradigm. According to this paradigm, such a formalism is assumed to accept convenient representations of input objects (messages) and produce, as output, adequate interpretations of these objects (which will be referred as *actions*), consistent with *a priori* knowledge (embodied as the conversation plan). By representing received information as sentences in an input language, and the actions to be carried out as a response to the interpretation of this information, as an output language, we propose as a formal device to accomplish this translation concept a *syntax-directed translation schema* (SDTS). In order to handle both the conversational contextual information (namely the agent knowledge base - which is common to all the conversations the agent maintains concurrently) and messages/actions parameters, we extend the previous formalism using attributed grammars.

2. SPEECH ACT THEORY

Speech act theory has been extensively used, formalised and extended within the fields of Computational Linguistics and Artificial Intelligence (AI) as a general model of communication between arbitrary agents, either human or artificial.

The main point of this theory (Austin, 1962) is that speech acts, besides being physical utterances (locutions), are mainly full-fledged actions, which reflect the speaker's intentions (illocution part) and may have social consequences (performative part) both in the hearer and the social environment. Elementary speech acts are seen as F(P) where F is the so-called illocutionary force and P is the prepositional content. Searle (1969) has classified speech acts according to their illocutionary force in 4 classes: 1) Assertives, which are statements of fact; 2) Directives, which are commands, requests or suggestions; 3) Commissives, e.g. promises, which commit the speaker to a course of action and 4) Expressives, which convey axiological content.

2.1 Business process modelling and the Language-Action Perspective

Inspired in Speech Act Theory, Winograd and Flores (1986) proposed the basic conversation loop, which was later used as a basis for developing the concept of Action Workflow, which is also the name of the corresponding software tool developed by Action Technologies (1993).

The Action Workflow model, which is restricted to communicative action, is based on the notion of "work as a closed loop". Figure 1 depicts the four phases that constitute the loop: preparation, negotiation, performance and acceptance. Each loop represents an elementary transaction and organisational processes are represented as elementary transactions between customers and performers.

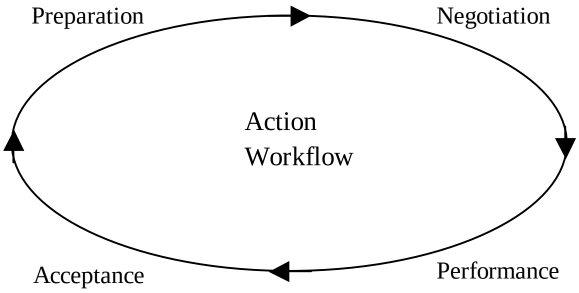


Figure 1: Action Workflow Loop (Medina-Mora et al., 1992).

This loop reflects a theory of organisation of work based on two roles and four phases. A transaction starts typically with a customer request. Then, after a performer's commitment and respective task execution, it ends up with customer satisfaction. The loop can be detailed to take into account sub-conversations involving the need of clarification, negotiation or other problems. In figure 2 is depicted a typical 'conversation for action'.

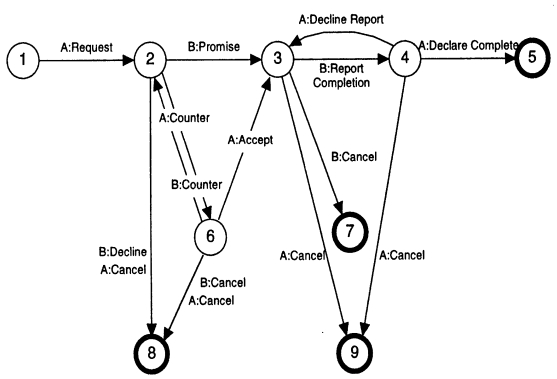


Figure 2: A conversation for action (Winograd and Flores, 1986).

Real-world business transactions have, almost always, complex domain-dependent structures, thus it is better to represent them using exception-based mechanisms instead of clobbering the main diagram with too many alternative behaviours.

Based on the theoretical work of Winograd (1988) several business process models have been developed, under the designation of 'Language Action Perspective' including Action Workflow (Medina-Mora et al., 1992), DEMO (Dietz, 1990; van Reijswoud, 1996; Barjis et al., 1999) and others (Verharen, 1997; Goldkhul et al., 1999).

A very important point to notice is that typical business process diagrams, as depicted in figure 2, show the analyst's perspective, not the agent designer's perspective. However, for our purposes, which are the design of autonomous, intelligent, agents, we need to provide to each agent conversation plans that it can use directly. This means that the overall business transaction must be decomposed into as many separate conversation plans as intervening agents.

2.2 Organisational Roles and Normative Conversation Plans

Conversation plans are normative concepts that are provided as templates for organisational behaviour. The behaviour of organisational agents is not specified individually but rather indirectly through organisational roles. Organisational agents are thus instantiations of organisational roles, inheriting their features.

A role is composed by (i) the functional description of its behavioural capabilities, i.e. the services it can provide; (ii) the set of policies that prescribe and constrain its behaviour, including both obligations and authorisations.

Policies can be represented as behavioural norms of the form:

Whenever <agent perceive event>
If <in conversation state> *then*
Agent adopts <attitude>

where attitude is typically a deontic action attitude, such as 'being obliged to do' or 'being permitted to do' some action.

Functional descriptions are either procedural descriptions of agent service routines or CPs. The latter are used to describe the interactive part of the functional capabilities of organisational roles. Conversations are instances of CPs at individual agent level.

3. SYNTAX-DIRECTED TRANSLATION SCHEMA AND ATTRIBUTED GRAMMARS

The power of grammatical formalisms stems from a threefold ability. They provide:

i) a compact model of the universe of the problem considered (representation),
ii) a generative capability, based on which new patterns can be generated,
iii) a mechanism for *recognition* by means of parsing algorithms.

Within the framework of formal languages' techniques for interpretation-based applications, the concept of *recognition* is replaced by the concept of t*ranslation*, which can be adequately formalised as a syntax-directed translation schema (Aho et al., 1972; Fred et al., 1996).

3.1 Formal Representation

A SDTS is a formal device that takes as input structural representations of objects (in the form of strings from an input language), and produces as output adequate interpretations of these objects, consistent with *a priori* knowledge (embodied as an underlying grammar), under the format of strings from an output language. The pair input string/output string is called a translation. Let Σ be an input alphabet and Δ an output alphabet. A translation from language $L_i \subseteq \Sigma^*$ to language $L_o \subseteq \Delta^*$ is a relation T from Σ to Δ such that the domain of T is L_i and the range of T is L_o. The formalism for specifying translations, SDTS, is basically a grammar in which translation elements are attached to each production:

Definition: A *syntax-directed translation schema* is a 5-tuple $\mathrm{T} = (V_N, \Sigma, \Delta, R, \sigma)$, where:

- V_N is a finite set of non-terminal symbols;
- Σ is a finite input alphabet, $\Sigma \cap V_N = \phi$;
- Δ is a finite output alphabet, $\Delta \cap V_N = \phi$;
- R is a finite set of rules of the form $A \to \alpha, \beta$, with $\alpha \in (V_N \cup \Sigma)^*$ and $\beta \in (V_N \cup \Delta)^*$
- σ is the start symbol.

The underlying grammar is, in general, context-free. When the domain knowledge admits a representation using finite state grammars, i.e. rules in R are of the form $A \to aB, bB$ or $A \to a, b$, with $a \in \Sigma, b \in \Delta$, $A, B \in V_N$, the SDTS can be represented as a graph. An example is shown in figure 3.

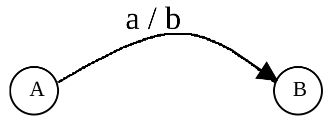


Figure 3: Graph representing the rule $A \to aB, bB$

In the context of conversations modelling, the input language corresponds to messages received by an agent; the output language consists of actions to be performed as a response to the interpretation of these messages. While conversations evolve, paths along these graphs are produced. In order to cope with parametrical messages (messages in predefined classes, the particular content being expressed in terms of a set of attributes or parameters), and aiming at including the agent information status into behavioural decisions, we extend the SDTS formalism with attributes. In this approach, attributes are introduced to the primitives, and semantic rules define the way these primitives are to be evaluated and propagated. This hybrid formalism, hereafter referred as *attributed-SDTS*, allows the reduction of syntactic complexity, by the addition of semantic rules in a fashion similar to the semantic-syntactic approach proposed in (Fu, 1986) as attributed grammars.

In the following, two types of semantic rules will be used: (1) *constraint rules*: each semantic rule is used as a semantic constraint to indicate the applicability of the corresponding syntactic rule; (2) *parameter propagation rules:* to set/propagate attributes associated with symbols.

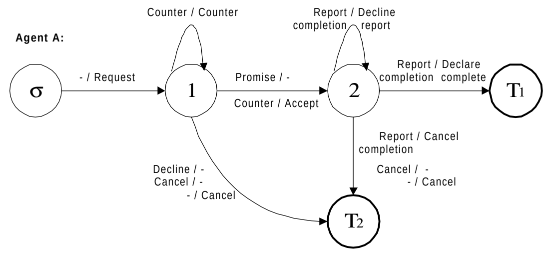
3.2 CPs in Business Processes

Since organisations are goal-directed multi-agent systems, whose behaviour is essentially based on business processes, below we describe how SDTS and the proposed extension – *attributed-SDTS* – can be used to represent the kind of conversations that occur in business process modelling, as described in section 2.

We use the business transaction model depicted in figure 2 to illustrate the applicability of the representation method described in the previous section (see figures 4a and 4b). Figure 4a represents the conversation plan of agent A and figure 4b represents the conversation plan of agent B. In these figures, nodes T1 and T2 represent terminal nodes for successful and unsuccessful transactions, respectively. Nodes 1 and 2 on the first graph (denoted by N_1,N_2 in the rules) correspond to waiting for response and waiting for completion states, respectively. On the graph for agent B, node 3 corresponds to a negotiation state. λ denotes the null string.

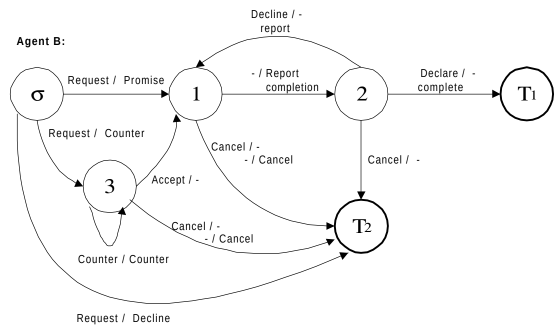
According to the proposed attributed-SDTS formalism, a set of attributes is associated with both terminal symbols and non-terminal symbols. For instance, attributes CC (current conversation), S (sender), R (receiver) are defined at the σ-rule level and inherited thereafter by terminal and non-terminal symbols. Message content (C) is instantiated at the message level and inherited by non-terminal symbols. A special attribute, *E*, (of Boolean type) is defined for non-terminal symbols, representing the applicability of the rule, based on an evaluation function that takes into account both the conversation state and the agent state.

Given a syntactic rule of the form $N_i \rightarrow aN_j, bN_j$ the following semantic rules are defined:



$$
\begin{array}{lll}
\sigma & \rightarrow & \lambda N_1, Request\ N_1 \\
N_1 & \rightarrow & Counter\ N_1, Counter\ N_1 \quad | \quad Counter\ N_2, Accept\ N_2 \\
 & & Promise\ N_2, \lambda N_2 \quad | \quad Decline, \lambda \\
 & & Cancel, \lambda \quad | \quad \lambda, Cancel \\
N_2 & \rightarrow & \lambda, Cancel \quad | \quad Report\_compl\ N_2, Decline\_report\ N_2 \\
 & & Cancel, \lambda \\
 & & Report\_compl, Cancel \quad | \quad Report\_compl, Declare\_complete
\end{array}
$$

Figure 4a: Graph and syntactic rules of the SDTS



$$
\begin{array}{lll}
\sigma & \rightarrow & Request\ N_1, Promise\ N_1 \quad | \quad Request\ N_3, Counter\ N_3 \\
 & & Request, Decline \\
N_1 & \rightarrow & \lambda N_2, Report\_completion\ N_2 \\
 & & Cancel, \lambda \quad | \quad \lambda, Cancel \\
N_2 & \rightarrow & Declare\_complete, \lambda \quad | \quad Decline\_report\ N_1, \lambda N_1 \\
N_3 & \rightarrow & Counter\ N_3, Counter\ N_3 \quad | \quad Accept\ N_1, \lambda N_1 \\
 & & Cancel, \lambda \quad | \quad \lambda, Cancel
\end{array}
$$

Figure 4b: Graph and syntactic rules of the SDTS

$$
\begin{aligned}
&CC(N_j) = CC(N_i) \\
&S(N_j) = S(N_i) \\
&R(N_j) = R(N_i) \\
&C(N_j) = C(a) \\
&E(N_i) = f(a, C(a), agent\_status)
\end{aligned}
$$

In the above semantic rules, the last one is a constraint rule, of Boolean type, that evaluates the current input and the agent state, to decide on the applicability of the corresponding syntactic rule. The other ones are parameter propagation rules.

4. IMPLEMENTATION

The communication architecture has 3 layers of communication, as indicated in figure 5: (i) a message layer, that resolves all aspects related to the message routing and transporting, independently of its illocutionary point or content; (ii) a conversation layer, that recognizes the message illocutionary point and manages conversation plans and conversations accordingly; and (iii) an agent layer where the message content is processed and the coherence of the agent multiple conversations is managed.

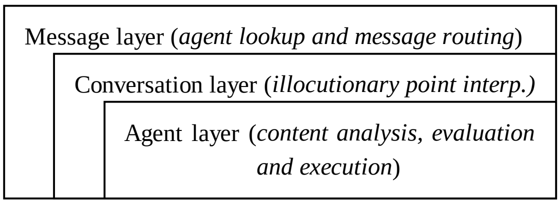


Figure 5: Communication layers.

We use the JINI environment for implementing the transport layer and JESS (the Java Expert System Shell) as a rule-based environment for representing conversation plans at the agent layer.

In order to be able to manage simultaneous conversations with different agents, we need to create, inside each agent, a conversation thread, which is implemented as a Java thread. This thread holds an inference engine of its own and a local knowledge base where exactly one role has been loaded. However, although each conversation thread has an individual state, all conversations share the agent “mind”, i.e. a common knowledge base, which must be consulted before committing any of the agent resources. The communicative agent architecture is depicted in figure 6.

5. CONCLUSIONS

In this paper we addressed the problem of conversation modelling in an environment of collaborative multi-agent systems. Although inspired in the language-action perspective to business process modelling, we have taken the designer’s perspective, instead of the analyst’s perspective. To enable the design of autonomous, intelligent, agents, overall business transactions must be decomposed into separate conversation plans, and specific conversation plans must be assigned to each agent, according to its organisational role(s).

We presented a formal method for conversation representation that is inspired in syntactic pattern recognition methods, specifically syntax-directed translation schemes. In order to model conversation parameters and global evaluation functions, a hybrid formalism was proposed that extended

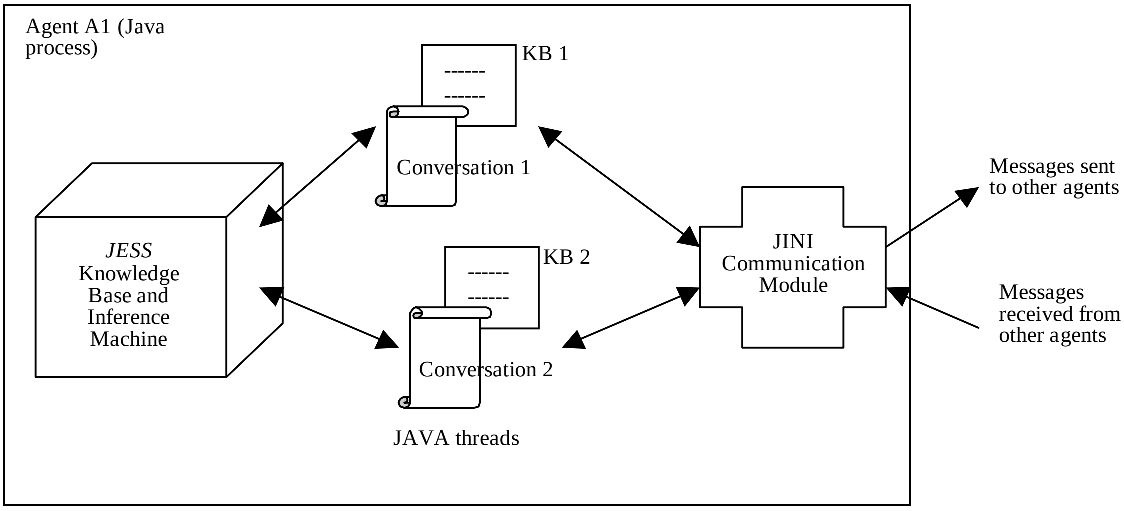


Figure 6: Communicative Agent Architecture.

the SDTS using attributed grammars. This method has a clear semantics that can be easily given a declarative implementation, thus becoming flexible enough to accommodate on-line extensions and exception handling.

References

Aho, A., J. Ullman, 1972. The Theory of Parsing, Translation and Compiling. Prentice-Hall.

Austin, J, 1962. How to do things with words. Harvard University Press, Cambridge, MA.

Barbuceanu, M., and M. Fox, 1997. Integrating Communicative Action, Conversations and Decision Theory to Coordinate Agents. *Proc. of Autonomous Agents'97*, 47-58, Marina Del Rey.

Barjis, J. and J. Dietz, 1999. Business Process Modeling and Analysis using GERT Networks. *In: Proceedings of ICEIS'99 - International Conference on Enterprise Information Systems*. Setúbal, pp. 748-756.

Dietz, J. (1990) A Communication Oriented Approach to Conceptual Systems Modeling. In: *Proceedings of the 1990 Working Conference on Dynamic Modelling of Information Systems*. Noordwijkerbout, pp. 37-60.

Filipe, J., K. Liu and B. Sharp, 1999. A Semiotic Approach to Organisational Role Modelling using Intelligent Agents, *2nd International Workshop on Organisational Semiotics,* Almelo, Holland.

Filipe, J., K. Liu and B. Sharp, 2000. Organisation Simulation Based on Normative Knowledge and Role Modelling, *AAAI Spring Symposium'2000*, Stanford, USA.

Fred, A. and J. Leitão, 1996. Parsing in attributed grammars: application to automatic sleep analysis. In: *Proc. of the Int'l Conference on Acoustics, Speech and Signal Processing, ICASSP 96*, Atlanta, pp 3466-3469.

Fu, K., 1986. A step towards unification of syntactic and statistical pattern recognition. *IEEE Trans. Pattern Analysis. and Machine Intelligence*, PAMI-8, pp. 398-404.

Goldkhul, G. and A. Rostlinger, 1999. Expanding the Scope – From Language Action to Generic Practice. In: *Proc. of the 4th Int'l Workshop on Language Action Perspective on Communication Modelling*, Copenhagen, pp.15-28.

Habermas, J., 1984. The Theory of Communicative Action: Reason and Rationalization of Society. Polity Press, Cambridge.

Labrou, Y., 1997. Semantics for An Agent Communication Language, Ph.D. thesis. University of Maryland, USA.

Medina-Mora, R., T. Winograd, R. Flores and F. Flores, 1992. The Action Workflow Approach to Workflow Management Technology. In: *Proceedings of the 4th Conference on Computer Supported Cooperative Work*. ACM, New York, pp. 281-288.

Searle, J., 1969. Speech Acts: An Essay in the Philosophy of Language. Cambridge University Press, Cambridge.

Singh, M., 1997. Commitments Among Autonomous Agents in Information-Rich Environments. *8th European Workshop on Modelling Autonomous Agents in a Multi-Agent World,* Roneby, Sweden.

van Reijswoud, V., 1996. The Structure of Business Communication: Theory, Model and Application. Ph.D. thesis, Delft University, Holland.

Verharen, E., 1997. A Language-Action Perspective on the Design of Cooperative Information Systems, Ph.D. thesis, Tilburg University, Holland.

Winograd T. and F. Flores, 1986. Understanding Computers and Cognition: a New Foundation for Design. Ablex, Norwood NJ.

Winograd, T., 1988. A Language-Action Perspective on the Design of Cooperative Work. In: Computer Supported Cooperative Work: A Book of Readings, I.Greif (Ed.), Morgan Kaufman.