A Multi-perspective Methodology for Modelling Inter-enterprise Business Processes

Kuldar Taveter[1,2] and Gerd Wagner[3]

[1] VTT Information Technology (Technical Research Centre of Finland),
P.O.Box 1201, FIN-02044 VTT, Finland
Kuldar.Taveter@vtt.fi
[2] Tallinn Technical University, Department of Informatics,
Raja 15, 12618 Tallinn, Estoniaa
[3] Eindhoven University of Technology, Faculty of Technology Management,
P.O. Box 513, 5600 MB Eindhoven, The Netherlands,
G.Wagner@tm.tue.nl
http://tmitwww.tm.tue.nl/staff/gwagner

Abstract. We propose an agent-oriented methodology for modelling business processes between enterprises that consists of the steps of analysis and design. The analysis starts by modelling intentional dependencies between the actors of the problem domain at hand, and continues by modelling intentional relationships that are internal to the actors, such as task decomposition links. In the design, intentional dependencies between the actors are transformed to commitment-based models of interactions between the agents. Models of tasks performed by the actors, obtained as a result of means-ends analysis, are transformed to activities and reaction rules, defining the behaviours of agents. The methodology is evaluated by using the case study of an electronic advertising process in newspapers.

1 Introduction

In information systems engineering, there is a gap between descriptive models needed at the stage of analyzing the problem domain at hand and prescriptive models needed later on at the stage of designing the information system for the problem domain. The gap is there because the notions used in modelling problem domains and information systems are too different. The new emerging paradigm of ***agent-orientation*** helps to bridge this gap by enabling to use the same terms like *beliefs*, *perceptions*, *memory*, *goals*, *commitments*, etc. for humans, institutions, and artificial (e.g., software-controlled)systems. According to this paradigm, communication between different artificial (software-controlled) systems and between systems, institutions, and humans is understood as communication between agents. Since the paradigm of agent-orientation promotes *autonomous action and decision-making*, it is highly relevant for modelling and implementing business processes [16, 20] involving different enterprises with their respective information systems.

H. Arisawa and Y. Kambayashi (Eds.): ER 2001 Workshops, LNCS 2465, pp. 403-416, 2002.

2 The Methodology of Agent-Oriented Modelling

2.1 The Modelling Process

We propose an agent-oriented methodology for modelling business processes between enterprises that consists of the steps of analysis and design. For the step of analysis, we make use of the ***i**** modelling framework proposed in [21, 25]. The analysis starts by modelling intentional dependencies between the actors of the problem domain at hand, and continues by modelling intentional relationships that are internal to the actors, such as task decomposition links.
For the step of design, we use the Agent-Object-Relationship (AOR) modelling proposed in [2, 8]. In the design, intentional dependencies between the actors are transformed to commitment-based models of interactions between the agents. Models of tasks performed by the actors, obtained as a result of means-ends analysis, are transformed to models of activities and reaction rules, defining the behaviours of agents.

2.2 Analysis of the Problem Domain by *i**

For the analysis of the problem domain at hand, we make use of the ***i**** framework proposed in [21, 25]. The ***i**** (which stands for "distributed intentionality") framework provides understanding of the motivation of social actors that depend on each other for goals to be achieved, tasks to be performed, and resources to be furnished. The framework consists of a **Strategic Dependency** (**SD**) model and a **Strategic Rationale** (**SR**) model. The SD model provides an intentional description of a (business) process in terms of a network of dependency relationships among actors. The SR model provides an intentional description of a (business) process in terms of process elements and the rationales behind them [21].

Analysis of Dependencies. The SD model consists of a set of nodes and links. Each node represents an actor, and each link between two actors indicates that one actor depends on the other for something in order that the former may attain some goal. The depending actor is called the **depender**, and the actor who is depended upon the **dependee**. The object around which the dependency relationship centres is called the **dependum**. An **actor** is an active entity that carries out actions to achieve goals by exercising its knowhow. The term *actor* is used to refer generically to any unit to which intentional dependencies can be ascribed.
Four types of dependencies are distinguished among actors, based on the type of dependum. In a **goal dependency**, a depender depends on the dependee to bring about a certain state in the world. The dependee is given the freedom to choose how to do it. In a **task dependency**, a depender depends on the dependee to carry out an activity. A task dependency specifies *how* the task is to be performed, but not *why*. In a **resource dependency**, the depender depends on the dependee for the availability of an entity (physical or informational). By establishing this dependency, the depender gains the ability to use this entity as a resource. In a **softgoal dependency**, a depender depends

on the dependee to perform some task that meets a *softgoal*. The meaning of the softgoal is not clear-cut. It is specified in terms of the methods that are chosen in the course of pursuing the goal.

The intentional dependencies of the domain of newspaper advertising, distilled from [12], are represented in Figure 1. The figure includes five actor types: the *Advertiser*, *Media Agency*, *Publication*, *Artwork Designer*, and *Artwork Producer*. The *Advertiser* depends on the *Media Agency* to have an advertising campaign performed. The *Advertiser* is only concerned about the outcome of the campaign and doesn't care *how* exactly the campaign is organized by the *Media Agency*. Therefore the dependency "*Campaign performed*" would be appropriately modelled as a goal dependency.

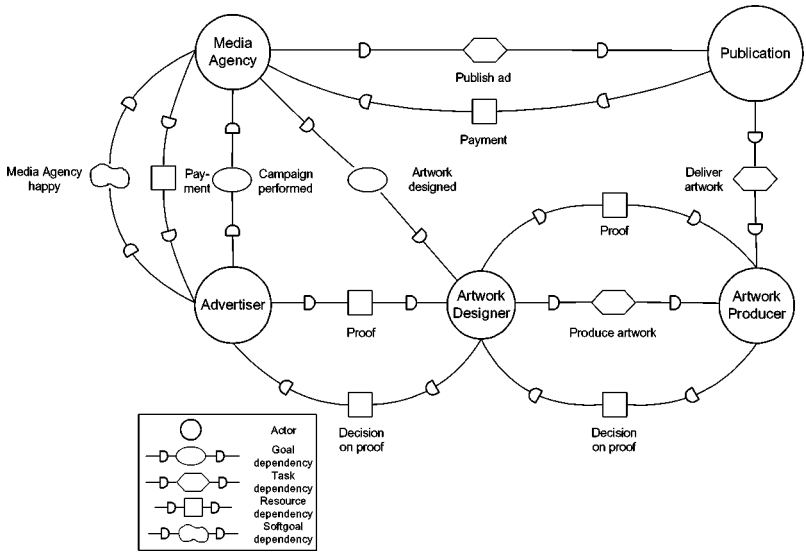

Fig. 1. The Strategic Dependency Model for the domain of newspaper advertising

Another goal dependency in Figure 1 is that the *Media Agency* depends on the *Artwork Designer* to have an artwork designed. Again, the *Media Agency* wants to have the artwork designed and doesn't care *how* it is done. Moreover, due to the artistic nature of designing an artwork, it is not even possible to provide the *Artwork Designer* with precise instructions how the artwork should be designed. As an example of a task dependency, the *Media Agency* depends on the *Publication* to have the ad published. Since the *Media Agency* wants to have the ad published according to the well-defined instructions established by it, the dependency "*Publish ad*" would be appropriately modelled as a task dependency. Another example of a task dependency is the dependency "*Produce artwork*" between the *Artwork Designer* and *Artwork Producer* where the *Artwork Designer* wants to have the artwork produced exactly according to the instructions provided by it. The dependency "*Deliver artwork*" between the *Publication* and *Artwork Producer* is naturally also modelled as a task dependency.

The dependencies of the *Media Agency* and *Publication* on respectively the *Advertiser* and *Media Agency* for a payment are modelled as resource dependencies where the dependum is an amount of money.

The *Advertiser* depends on the *Artwork Designer* and the *Artwork Designer* on the *Artwork Producer* for a proof of the artwork to be designed or produced. The *Artwork Designer* and *Artwork Producer*, in turn, depend on the *Advertiser* and *Artwork Designer*, respectively, for a decision on the proof. These dependencies are expressed as resource dependencies. The dependums here are proofs and decisions on the proofs.

Since the *Media Agency* acts as a customer for the *Publication*, there is a softgoal dependency "*Media Agency happy*" between the *Publication* and *Media Agency*.

Means-Ends Analysis. The SR model provides an intentional description of a (business) process in terms of process elements and the rationales behind them. While the SD model represents only external relationships between actors, the corresponding SR model describes intentional relationships that are internal to actors, such as task decomposition links.

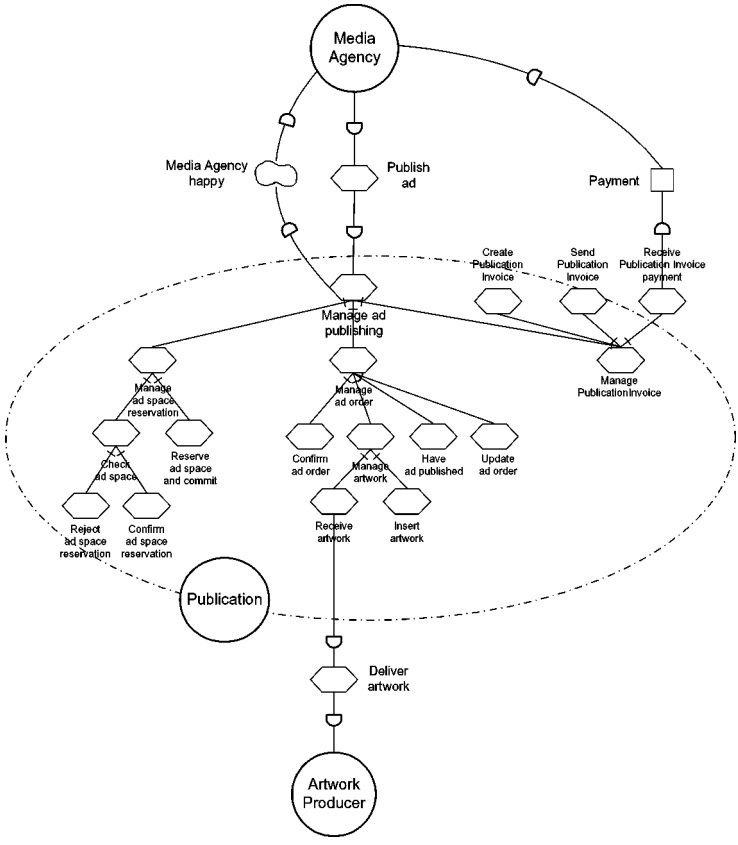

Fig. 2. The Strategic Rationale Model for the *Publication*

The SR model is a graph, with several types of nodes and links. There are four types of nodes, based on the same distinction made for dependum types in the SD model – goal, task, resource, and softgoal. A **goal** is a condition or state of affairs in the world that the actor would like to achieve. A **task** specifies a particular way of doing something. A **resource** is a physical or informational entity to be used or produced by the actor. A **softgoal** is a condition in the world which the actor would like to achieve, but the criteria for the condition being achieved are not sharply defined *a priori*, and are subject to interpretation.
Only tasks are present in Figure 2. A task node is linked to its component nodes by **task decomposition links**. Tasks can also connect up with dependency links in SD model(s), when the reasoning goes beyond an actor's boundary.
The SR model for the *Publication*, created based on [12], is represented in Figure 2. The model shows that the *Publication* is able to fulfill the task dependency "*Publish ad*" that the *Media Agency* depends on by running the internal task "*Manage ad publishing*". The task "*Manage ad publishing*" consists of the subtasks "*Manage ad space reservation*", "*Manage ad order*", and "*Manage Publication Invoice*". These subtasks, in turn, consist of other subtasks. Some of the subtasks have alternative subtasks like "*Reject ad space reservation*" and "*Confirm ad space reservation*" of the subtask "*Check ad space*", but their alternativeness cannot be represented by means of ***i****. Therefore it is deferred to section 2.3 where it is done by reaction rules of agents. The task "*Manage* ad publishing" and the subtasks "*Receive artwork*" and "*Receive Publication Invoice payment*" respectively connect up with the softgoal dependency "*Media Agency happy*", task dependency "*Deliver artwork*", and resource dependency "*Payment*".

2.3 Design by Agent-Object-Relationship Modelling

Since intentional dependencies between actors of ***i**** imply inter-agent interactions and process elements internal to actors such as tasks and goals imply activities of agents, we propose to model interactions and activities by Agent-Object-Relationship (AOR) Modelling for designing an inter-enterprise information system. In this section, we will first present an overview of AOR Modelling and will then describe mapping from the SD and SR models of ***i****, which we used at the stage of analyzing the problem domain, to the modelling abstractions of AOR modelling.

Principles of Agent-Object-Relationship Modelling. Agent-Object-Relationship diagrams were proposed in [2, 8] as an agent-oriented extension of Entity-Relationship-diagrams, or UML-style class diagrams. In order to capture more semantics of the dynamic and deontic aspects of organizations involved in business processes and their information systems, such as the events and actions related to the ongoing business processes, it is proposed to make an ontological distinction between active and passive entities, that is, between *agents* and *ordinary objects*. AOR modelling suggests that the semantics of business transactions can be more adequately captured if the specific *business agents* associated with the involved events and actions are explicitly represented in organizational as well as in inter-organizational information systems in addition to passive *business objects*.

Entity types. In AOR modelling, an entity is either an *agent*, an *event*, an *action*, a *claim*, a *commitment*, or an ordinary *object*. An organization is viewed as a complex ***institutional agent***, being involved in a number of interactions with external agents and consisting of a number of internal agents that perceive events and perform actions on behalf of it. ***Internal agents*** may be humans, artificial agents (such as software agents, agentified information systems, robots, or agentified embedded systems), or institutional agents (such as organization units).
The entity types listed above can be viewed and represented as the corresponding stereotypes of UML [28]: *<<agent>>*, *<<event>>*, *<<action>>*, *<<activity>>*, *<<claim>>*, *<<commitment>>*, and *<<object>>*.
As usual, entity types are visually represented by rectangles while relationship types are represented by connection lines (possibly with crows feet endings in order to indicate multiplicity). While an *object type* is visualized as an ordinary rectangle, an *agent type* is graphically rendered as a rectangle with rounded corners. An internal agent type is visualized by such a rectangle with a dashed line drawn within the institutional agent rectangle it belongs to (like Department in Figure 3). An instance of an agent type is distinguished from an agent type by underlining its name (like the Publication Secretary in Figure 3).

Actions and events. In a business domain, there are various types of actions performed by agents, and there are various types of state changes, including the progression of time, that occur in the environment of the agents. For an external observer, both actions and environmental state changes constitute events. In the internal perspective of an agent that acts in the business domain, only the actions of other agents count as events.
Actions create events, but not all events are created by actions. Those events that are created by actions, such as delivering a product to a customer, are called **action events**. Examples of business events that are not created by actions are the fall of a particular stock value below a certain threshold, or a timeout in an auction. Actions make up activities. While an action happens at a time point (i.e., it is immediate), an **activity** is being performed during a time interval (i.e., it has duration), and consists of a set of actions. We distinguish between a ***human activity*** and an ***automated activity*** performed by e.g. a software agent.
We make a distinction between *communicative* and *non-communicative* actions and events. Many typical business events, such as receiving a purchase order or a sales quotation, are communication events. The expressions *receiving a message* and *sending a message* may be considered to be synonyms of *perceiving a communication event* and *performing a communication act*.
Business communication may be viewed as ***asynchronous*** point-to-point message passing. The feature of asynchronous communication is a fundamental one in agent-oriented modelling that makes the latter especially suitable for modelling inter-enterprise business processes. This feature distinguishes agent-oriented modelling from other modelling techniques requiring synchronous communication like e.g. Role Activity Diagrams [29].
As opposed to the low-level (and rather technical) concept of messages in object-oriented programming, AOR modelling assumes the high-level semantics of speech-act-based *Agent Communication Language (ACL)* messages (see [18, 19]).

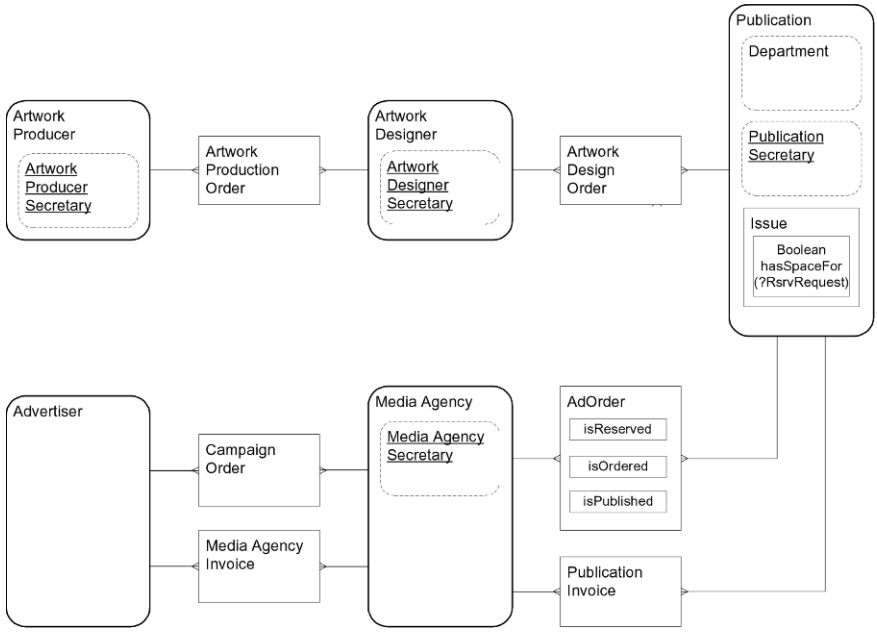

Fig. 3. The agent diagram of the domain of newspaper advertising. The agents include the respective (human) internal agents Media Agency Secretary, Publication Secretary, Artwork Designer Secretary, and Artwork Producer Secretary, and (instances of) the (institutional) internal agent type Department which is not further specified. The agents have knowledge/information about shared objects of certain types. An entity of the class AdOrder can have the status isReserved, isOrdered, or isPublished. The object type Issue has the intensional predicate hasSpaceFor (?RsrvRequest).

Commitments and claims. Commitments are fundamental components of business interaction processes. Representing and processing commitments and claims in information systems explicitly helps to achieve coherent behaviour in interaction processes. In [26], the social dimension of coherent behaviour is emphasized, and a **social commitment C**(x, y, G, p) is defined as a relation between the ***debtor*** x, the ***creditor*** y, and the ***discharge condition*** p that the debtor is committed to, in the scope of the ***social context*** G. The context is a group that *contains* the participating agents and has some sovereignty over its members. In the example of newspaper advertising, the context group imposes certain external constraints on the operation of media agencies and publications, like the ones stating how claims and disputes should be settled between a media agency and a newspaper.

A commitment c = **C**(x, y, G, p) is **base-level** if p does not refer to any other commitments; c is a **metacommitment** if p refers to a base-level commitment. Metacommitments create a society where the metacommitments are the norms of this society [26]. In our case study of newspaper advertising, the society consists of agents of the types shown in Figure 1.

Commitments typically arise from certain communication acts. For instance, sending a sales quotation to a customer commits the vendor to reserve adequate stocks of the

quoted item for some time. Likewise, acknowledging a sales order implies the creation of a commitment to deliver the ordered items on or before the specified delivery date.
There are two kinds of commitments: commitments to do an action and commitments to see to it that some condition holds. The former are called ***to-do*** commitments, and the latter ***see-to-it-that*** commitments. A *to-do* commitment is discharged by performing the corrresponding action, while a *see-to-it-that* commitment is discharged by performing an action or activity whose state changing effect implies the discharge condition.
In the perspective of a particular agent, commitments of other agents are viewed as ***claims*** against them.

Modelling Business Processes by Interaction Frames. The ***i**** framework enables to describe and analyze commitments that exist between the actors of a problem domain. We are also interested in how commitments arise and evolve, i.e. in prescriptive models of commitments. According to [21], an additional step needs to be taken to arrive at prescriptive requirements specifications from ***i****. We propose to achieve a more prescriptive stance of modelling through representing intentional dependencies as metacommitments by using the notion of social commitment.
For a goal dependency in the ***i**** framework, the condition *p* is an assertion *achieved*(*g*) representing the goal *g* to be achieved by an agent; for a task dependency on task *t*, *p* is *done*(*t*); for a resource dependency on resource *r*, *p* is *avail*(*r*) [21]. For mapping from ***i**** to AOR, *we represent an intentional dependency as a* ***metacommitment*** *of the depender towards the dependee to create (a) commitment(s) to satisfy the condition p*. For example, the goal dependency "*Campaign performed*" in Figure 1 of the *Advertiser* on the *Media Agency* can be represented as a *metacommitment* of the *Media Agency* towards the *Advertiser* by which the *Media Agency* commits upon receiving an order for campaign from the *Advertiser* to create a commitment towards the *Advertiser* to achieve the goal "*Campaign performed*". *We model business processes through lifecycles of commitments, governed by metacommitments, by using commitment protocols* [17, 24]. After that we capture lifecycles of commitments by using interaction frames of external AOR models. In an **external AOR model**, we adopt the view of an external observer who is observing the (prototypical) agents and their interactions in the problem domain under consideration. Typically, an external AOR model will have a *focus*, that is an agent, or a group of agents, for which we would like to develop an interaction model. In the view of an external observer, all actions of agents are at the same time also events, and commitments are also claims, exactly like two sides of the same coin. Therefore, an external AOR model contains, besides the agent and object types of interest, the action event types and commitment/claim types that are needed to describe the interaction between the focus agent(s) and the other types of agents.
An **interaction frame diagram**, in an external AOR model, provides a static picture of possible interactions and evolvement of commitments/claims within them. The interaction frame diagram, covering the business process type of organizing an advertising campaign, is depicted in Figure 4.

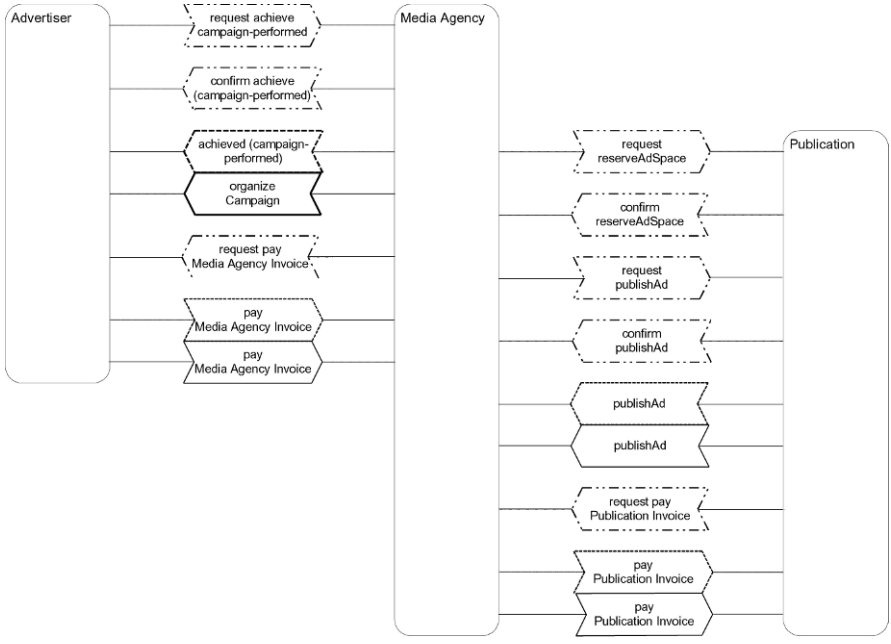

Fig. 4. The interaction frame for the business process type of organizing an advertising campaign with three focus agents. The frame starts with the formation of the *see-to-it-that* commitment achieved(campaign-performed) between agents of the types Advertiser and Media Agency. The occurrence of an action event of the type publishAd between Media Agency and Publication is preceded by the formation and refinement of the corresponding commitment/ claim of the type publishAd, as a consequence of the exchange of communicative action events of the types request reserveAdSpace, confirm reserveAdSpace, request publishAd, and confirm publishAd. The occurrence of a communicative action event of the type request pay Publication Invoice gives rise to a commitment/claim of the type pay Publication Invoice which is satisfied by the occurrence of an action event of the type pay Publication Invoice. The latter also concludes the activity organizeCampaign, as its discharge condition achieved (campaign-performed)becomes true. Finally, Media Agency sends a Media Agency Invoice to Advertiser,the latter commits and pays the Media Agency Invoice.

In the graphical notation of AOR modelling, *to-do* commitments are coupled with the action events satisfying them. For example, in Figure 4 the commitment of the publishAd type is coupled with the action event of the corresponding type. A *see-to-it-that* commitment is coupled with the activity completing of which makes the discharge condition *p* of the commitment true. For example, in Figure 4 the *see-to-it-that* commitment achieved(campaign-performed) is satisfied by the activity organizeCampaign which consists of several action events related to having the ad(s) published in one or more publications.

Modelling Agent Behaviours by Activity Diagrams. The behaviour of a Knowledge-Perception-Memory-Commitment agent introduced in [4, 10] is encoded by a set of ***reaction rules***. A reaction rule does not have any counterpart in UML [28], but it can be viewed as a construct complementing UML.

Reactive behaviour of agents is the most dominant one in a business domain. Reactive behaviour may also be combined with proactive behaviour, but this is not the topic of the present paper.

Each task identified by means-ends analysis (v. section 2.2) is mapped to one or more activities. An activity can be started in response to an event by means of a reaction rule. Once an activity is started, the agent is in the corresponding *activity state*. An activity can trigger reaction rules or subactivities through the implicit triggering event startOf(activity). Additionally, each activity is associated with another implicit triggering event endOf(activity) that can trigger other reaction rules and activities.

An activity may also have a ***goal***. For example, the goal of the activity organizeCampaign in Figure 4 is campaign-performed.

Activities are visualized as rectangles with rounded left and right sides, as shown in Figure 5. The triggering event startOf(activity) is represented by an empty circle with the outgoing arrow to the activity or reaction rule triggered by it. The triggering event endOf(activity) is modelled by drawing just the outgoing arrow from the activity to either the next activity or the reaction rule triggered by it.

A reaction rule is visualized as shown in the legend for Figure 5. If the composition of an activity is not specified, like in case of the activity "*Have ad published*", the incoming arrow representing a status condition, the outgoing arrow denoting a status change, and the outgoing connector to an action type are directly connected to the rectangle representing the activity.

In Figure 5, the activity "*Manage ad publishing*" performed by the Publication and its constituent activities "*Manage ad space reservation*" and "*Manage ad order*" are modelled on the basis of six reaction rules. Variables in the parameter list of a predicate are prefixed with a question mark. The activities "*Confirm ad order*" and "*Update ad size*" require an action by a human internal agent Publication Secretary and are therefore *human activities*, while all the other activities are to be performed by automated software agents and are thus *automated activities*.

R1: Upon receiving from the MediaAgency a request to reserve ad space, the activity "*Manage ad publishing*" is started.

R2: Upon the start of the subactivity "*Check ad space*", if the Publication has enough ad space on the date requested – determined by evaluating the intensional predicate Issue.hasSpaceFor(?RsrvRequest) – the Publication sends a confirmation to the Media Agency. Otherwise, if the Publication does not have sufficiently ad space on the date requested, the Publication sends a rejection to the Media Agency.

R3: Upon the start of the subactivity "*Reserve ad space and commit*", the Publication creates the corresponding ad space reservation (i.e. an instance of AdOrder with the status isReserved), and commits towards the Media Agency to publish the ad.

R4: Upon receiving an ad order from the Media Agency, the subactivity "*Manage ad order*" is started.

R5: Upon the approval of the ad order by the Publication Secretary, the Publication changes the status of the corresponding ad space reservation to isOrdered, and sends a confirmation to the Media Agency.

R6: Upon the insertion of the actual ad size of the ad published by the Publication Secretary, the corresponding object instance of the type AdSize is updated.

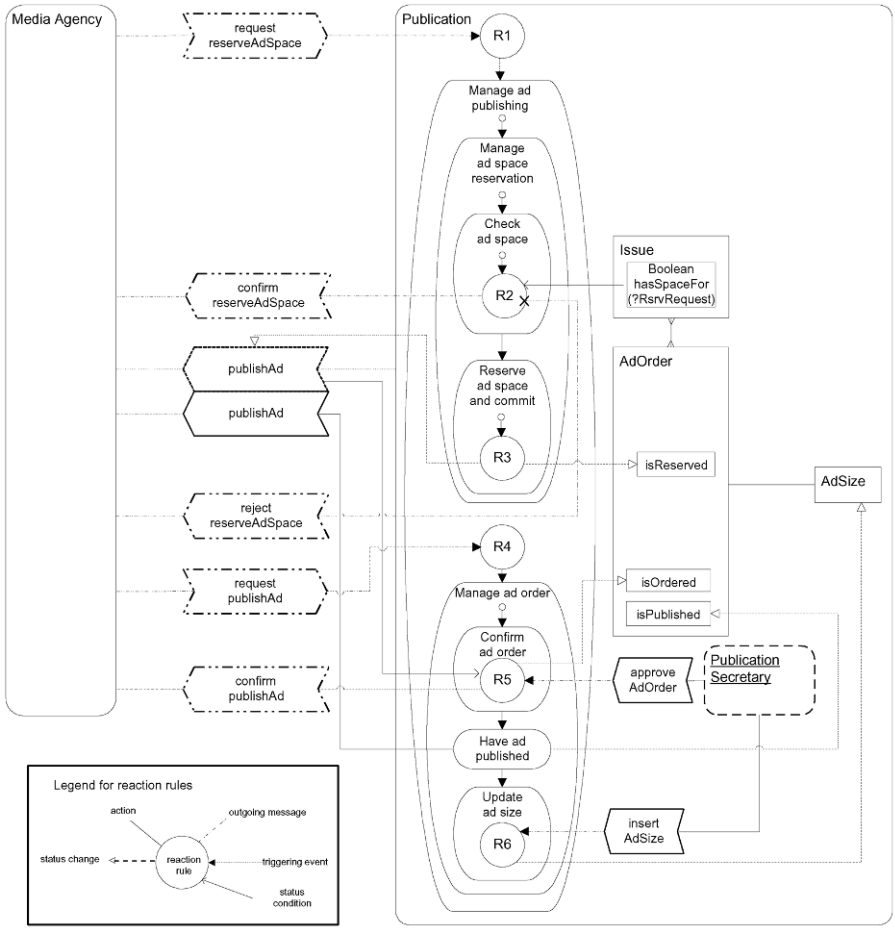

Fig. 5. Visualization of activities and reaction rules by an activity diagram.

3 Related Work

In the paper [22] a general methodology for agent-oriented analysis and design is presented. The methodology proposed deals with both the macro-level (societal) and the micro-level (agent) aspects of systems. In the analysis phase of the methodology, the *roles* in the system are identified and the patterns of interaction that occur in the system between various roles are recognized. The functionality of each role is defined by its liveness and safety responsibilities. *Liveness responsibilities* are those that say "something will be done", e.g. "whenever the coffee machine is empty, fill it up". *Safety responsibilities* relate to the absence of some undesirable condition arising, e.g. "the coffee stock should never be empty". In the design phase, the liveness and safety

responsibilities are respectively mapped to agents' *services* and *pre-* and *postconditions* on each service. The difference from our work is that the methodology proposed in [22] is a software engineering approach, while our approach is aimed at creating business information systems and therefore contains notions needed for it like intentional dependencies, goals, actions/events, and commitments/claims. The Tropos methodology described in [23] uses the models of ***i**** all along the lifecycle of an information system as a foundation to model both early and late requirements, architectural and detailed design, and implementation. An advantage of our approach compared to Tropos is that our approach moves with the actors and tasks identified at the stage of analysis to the stage of design where it distinguishes between human and automated activities, while in Tropos the activities modelled at design by UML [28] and AUML [27] are to be performed by software agents, and human activities disappear somewhere. Also, the target implementation model of Tropos is the BDI agent architecture [11, 14] which may be too restrictive and inefficient for implementing real-life information systems.

4 Conclusions and Future Work

The main contribution of our work is offering a methodology for business modelling and designing global information systems that bridges the gap between the perspectives of descriptive and prescriptive modelling by means of (meta)commitments. We propose to start engineering of information systems in support of inter-enterprise business processes from descriptive modelling of intentional dependencies between actors of the problem domain by SD models of ***i****, and to continue by analyzing means-ends reasoning of individual actors using SR models of ***i****. After that, the intentional dependencies are represented as metacommitments that serve as a basis for modelling lifecycles of first-order commitments by interaction frames of AOR modelling. We also suggest to refine the tasks identified by SR models by representing them as AOR activity diagrams which enables to identify the atomic actions together with their triggering events, and pre- and postconditions, and to transform all four to reaction rules of AOR modelling. Since reaction rule is a generic notion (e.g. triggers in relational databases [9] are reaction rules of a kind), reaction rules can be rather straightforwardly mapped to specific agent architectures and/or platforms like BDI [11, 14], vivid agents [10], AgentBuilder® [1], or the behaviour-based agent model of the JADE platform [3], compliant with the FIPA ACL [18]. Our approach thus enables to choose between different implementation architectures and platforms.

The notion of commitment also enables to capture various *business rules* [5, 6, 13] in a more natural way. This aspect of our work is addressed in [7] and [15]. We plan to add models for exception handling where exceptions are captured as violations of commitments. We also intend to find a suitable formalism for representing possible conflicts and inconsistencies in the agent's commitments, and for checking the agent's commitments on them in the precondition parts of reaction rules. Our future aims also include further *verification* and *validation* of our work.

References

1. AgentBuilder®, http://www.agentbuilder.com
2. Wagner, G.: Agent-Oriented Enterprise and Business Process Modeling. In: Proceedings of the First International Workshop on Enterprise Management and Resource Planning Systems (EMRPS'99), Venice, November 1999.
3. Bellifemine, F., Poggi, A., Rimassa, G.: Developing multi-agent systems with FIPA-compliant agent framework. Software – Practice and Experience **31** (2001) 103-128.
4. Wagner, G.: Foundations of Knowledge Systems with Applications to Databases and Agents. Kluwer Academic Publishers, Boston Dordrecht London (1998).
5. Ross, R. G.: The Business Rule Book: Classifying, Defining and Modeling Rules. Second Edition. Boston, Massachusetts, Database Research Group, Inc. (1997).
6. Defining Business Rules - What Are They Really? The Business Rules Group, formerly known as the GUIDE Business Rules Project, Final Report. Revision 1.3, July 2000. Prepared by D. Hay and K. A. Healy. Available at http://businessrulesgroup.org/first_paper/br01c0.htm
7. Taveter, K., Wagner, G.: Agent-Oriented Enterprise Modeling Based on Business Rules. In: Proceedings of the 20th International Conference on Conceptual Modeling (ER'2001), Yokohama, Japan, November 27-30, 2001. Springer-Verlag, Berlin Heidelberg New York (2001), forthcoming.
8. Wagner, G.: Agent-Oriented Analysis and Design of Organizational Information Systems. In: Barzdins, J., Caplinskas, A. (eds.): Databases and Information Systems. Proceedings of the Fourth IEEE International Baltic Workshop on Databases and Information Systems, 1–5 May 2000, Vilnius, Lithuania. Kluwer Academic Publishers, Boston Dordrecht London (2000).
9. Widom, J., Ceri, S. (eds.): Active Database Systems: Triggers and Rules for Advanced Database Processing. Morgan Kaufmann Publishers, Inc., San Francisco (1996).
10. Wagner, G.: Vivid Agents - How they Deliberate, How They React, How They Are Verified. Extended version of: Wagner, G.: A Logical And Operational Model of Scalable Knowledge- and Perception-Based Agents. In: Van de Velde, W., Perram. J. W. (eds.): Agents Breaking Away. Proceedings of MAAMAW'96. Lecture Notes in Artificial Intelligence, Vol. 1038. Springer-Verlag, Berlin Heidelberg New York (1996). Available At http://tmitwww.tm.tue.nl/staff/gwagner
11. Georgeff, M. P., Lansky, A.: Reactive reasoning and planning. In: Proceedings of the Sixth National Conference on Artificial Intelligence (AAAI-87), Seattle, Washington, USA (1987) 677–682.
12. Antikainen, H., Bäck, A.: Challenges of electronic advertising processes in newspapers. Technical report. VTT Information Technology, Espoo, Finland (2000).
13. Herbst, H.: Business Rule-Oriented Conceptual Modeling (Contributions to Management Science). Springer-Verlag Berlin Heidelberg New York (1997).
14. Huber, M. J.: Jam Agents in a Nutshell, http://members.home.net:80/marcush/IRS/
15. Taveter, K., Wagner, G.: Combining AOR Diagrams and Ross Business Rules' Diagrams for Enterprise Modeling. In: Proceedings of the Second International Bi-Conference Workshop on Agent-Oriented Information Systems (AOIS-2000), 5-6 June 2000, Stockholm (Sweden) and 30 July 2000, Austin (Texas, USA). iCue Publishing, Berlin (2000).
16. Hammer, M., Champy, J.: Reengineering the Corporation. New York, Harper Collins (1993).
17. Yolum, P., Singh, M.P.: Synthesizing Finite State Machines for Communication Protocols. North Carolina State University Technical Report TR-2001-06 (2001).
18. Foundation for Intelligent Physical Agents (FIPA), http://www.fipa.org
19. Knowledge Query and Manipulation Language (KQML), http://www.cs.umbc.edu/kqml/

20. Davenport, T. H.: Process Innovation: Reengineering Work through Information Technology. Harvard Business School Press (1992).
21. Yu, E.: Modelling Strategic Relationships for Process Reengineering. PhD thesis, Department of Computer Science, University of Toronto (1994).
22. Wooldridge, M., Jennings, N., Kinny, D.: The Gaia Methodology for Agent-Oriented Analysis and Design. Autonomous Agents and Multi-Agent Systems **3** (2000) 285-312. 23. Castro, J., Kolp, M., Mylopoulos, J.: A Requirements-Driven Development Methodology. In: Proceedings of the 13th International Conference on Advanced Information Systems Engineering (CAiSE'01), Interlaken, Switzerland, June 4-8, 2001. Springer-Verlag Berlin Heidelberg New York (2001).
24. Venkatraman, M., Singh, M. P.: Verifying Compliance with Commitment Protocols: Enabling Open Web-Based Multiagent Systems. Autonomous Agents and Multi-Agent Systems **3** (2000) 217-236.
25. Yu, E. S. K., Mylopoulos, J.: From E-R to 'A-R' - Modelling Strategic Actor Relationships for Business Process Reengineering. International Journal of Intelligent and Cooperative Information Systems **2/3** (1995) 125-144.
26. Singh, M.: An Ontology for Commitments in Multiagent Systems: Toward a Unification of Normative Concepts. Artificial Intelligence and Law **7** (1999) 97-113.
27. Odell, J., Van Dyke Parunak, H. Bernhard, B.: Representing Agent Interaction Protocols in UML. In: Proceedings of the First International Workshop on Agent-Oriented Software Engineering (AOSE-2000), Limerick, Ireland, June 2000.
28. Unified Modelling Language (UML), http://www.omg.org/uml/ 29. Ould, M. A.: Business processes: modelling and analysis for re-engineering and improvement. John Wiley & Sons, Chichester (1995).