Formal Modeling and Analysis of Organizations

Egon L. van den Broek[1], Catholijn M. Jonker[2], Alexei Sharpanskykh[1], Jan Treur[1], and pInar Yolum[3]

[1] Department of Artificial Intelligence, Vrije Universiteit Amsterdam, De Boelelaan 1081a, NL-1081 HV Amsterdam, The Netherlands
{egon, sharp, treur}@few.vu.nl
[2] NICI, Radboud University Nijmegen Montessorilaan 3, 6525 HR Nijmegen, The Netherlands
C.Jonker@nici.ru.nl
[3] Department of Computer Engineering, Bogazici University, TR-34342 Bebek, Istanbul, Turkey
pyolum@cmpe.boun.edu.tr

Abstract. A new, formal, role-based, framework for modeling and analyzing both real world and artificial organizations is introduced. It exploits static and dynamic properties of the organizational model and includes the (frequently ignored) environment. The transition is described from a generic framework of an organization to its deployed model and to the actual agent allocation. For verification and validation purposes, a set of dedicated techniques is introduced. Moreover, where most models can handle only two or three layered organizational structures, our framework can handle any arbitrary number of organizational layers. Henceforth, real-world organizations can be modeled and analyzed, as illustrated by a case study, within the DEAL project line.

1 Introduction

Organizations have proven to be a useful paradigm for analyzing and designing multi-agent systems (MAS) [5, 21, 22]. Representation of MAS as an organization consisting of roles and groups can tackle major drawbacks concerned with traditional multi-agent models; e.g., high complexity and poor predictability of dynamics in a system [5, 21]. We adopt a generic representation of organizations, abstracted from instances of real agents. As has been shown in [9], organizational structure can be used to limit the scope of interactions between agents, reduce or explicitly increase redundancy of a system, or formalize high-level system goals, of which a single agent may be not aware. Moreover, organizational research has recognized the advantages of agent-based models; e.g., for analysis of structure and dynamics of real organizations. However, formal theories, approaches, and tools for designing such models are rare. In this paper, we propose a new modeling approach for analyzing and formal modeling of real or artificial organizations (e.g., MAS).

In the next Section, main principles for modeling and analyzing organizations are discussed and related with the new modeling approach. In Section 3, the basic concepts used for specifying an organization model are introduced. Section 4 discusses how an organization model can be specified in a formal manner. In Section 5, a set of

O. Boissier et al. (Eds.): ANIREM and OOOP 2005, LNAI 3913, pp. 18 – 34, 2006.

dedicated validation and verification techniques are described. Section 6 presents a case study, which explains how the proposed approach can be applied for analyzing an organization from the area of logistics. It includes the introduction of a new technique for the graphical representation of organization models. The paper ends with a discussion in Section 7.

2 Principles for Modeling and Analyzing Organizations

Modern organizations are characterized by their complex structure, dense information flows, and incorporation of information technology. To a large extent, the underlying organization model is responsible for how efficiently and effectively organizations carry out their tasks. In literature, a range of theories and guidelines concerning the design of organizations are present [15, 17]. However, almost no operational theories or formal models exist. Scott [20] even stated that there are no general principles applicable to organizational design. In contrast, Minzberg proposed a set of guidelines for modeling any arbitrary organization [15]. These guidelines are applicable to mechanistic types of organizations, which represent systems of hierarchically linked job positions with clear responsibilities that use standard well-understood technology and operate in a relatively stable (possibly complex) environment. However, many modern organizations are characterized by a highly dynamic, constantly changing, organic structure and show a hardly identified, not formalized, non-linear behavior [16].

2.1 Two Perspectives

In this subsection, we will briefly discuss two perspectives from which organizations are analyzed. The first perspective emerges from social sciences and the second originates from computational organization theory and artificial intelligence.

In social science theories, the structure of organizations is frequently specified as informal or semi-formal graphical representations [15, 17]. They can provide a detailed organization structure at an abstract level. However, such approaches lack the means to represent the more detailed dynamics and to relate them to the structures present.

From computational organization theory and artificial intelligence, approaches have been developed that are able to capture both structural and dynamic aspects of organizations. However, usually they describe organization models, using only two or three levels of abstraction; i.e., the level of an individual role, the level of a group composed of roles, and the overall organization level, as in GAIA [22], MOISE [7] (extended to S-MOISE+ [11]), MOCA [1], and OperA [3]. In contrast, multiple levels and relations need to be described for the representation of complex hierarchical structures of modern organizations; e.g., mechanistic type of organizations [17].

Some models (e.g., ISLANDER [4], OperA [3]) consider organizations as electronic institutions; i.e., norms and global rules that govern an organization are explicitly defined. However, in many modern organic organizations with much individual autonomy, the normative aspects do not play a central role and are of minor importance for the prosperity of an organization.

Independent of the previous distinction in approaches, the importance of explicit modeling of interactions between agents and the environment is recognized [3, 18]. This is of importance since the environment plays a crucial role in the functioning of organizations. Moreover, for modeling in general, verification and validation of the models used or generated is of the utmost importance. This is no different for modeling organizations. However, this aspect of modeling organizations is frequently ignored; two of the exceptions are TROPOS [2] and ISLANDER [4].

2.2 A New Perspective

In this paper, we propose an approach for formal specification of organizations. To this end, it is highly suitable for specifying mechanistic types of organizations; i.e., machine and professional bureaucracy and divisionalized forms of organizations. Furthermore, this approach can also be applied for modeling organic types of organizations, when extended with organizational change techniques.

The proposed, formal approach can capture both structural and dynamic aspects of the organization and, subsequently, has four advantages:

(1) Representation of organization structure (including specifications of actors (or roles), relations between them, and information flows).
(2) The means for simulations of different scenarios on the basis of a model and observing their results.
(3) Organization analysis by means of verifying static and dynamic properties against (formalized) empirical data, taken from real organizations, or against simulated scenarios.
(4) Diagnosis of inconsistencies, redundancies, and errors in structure and functioning (e.g., with regard to organizational performance indicators) of real organizations and providing recommendations for their improvement.

In the proposed model, organizations are specified as composite roles that can be refined. The refined structures consist of (interacting) roles, representing as many aggregation levels as needed. Moreover, global normative aspects of an organization are considered as static and dynamic properties of the role, defined at the highest abstraction level, which represents the whole organization, without recognizing them as special notions and placing them on top of an organization. In addition, the environment is considered as a special component of an organization model.

The modeling method introduced in this paper incorporates two types of verification and validation techniques: role-centered and agent-centered, as will be discussed in Section 5. However, the introduction of these techniques is preceded by the introduction of the model itself in the next section and its formal specification in Section 4.

3 Organization Modeling Concepts

In this section, the concepts are introduced on which the organization modeling approach is founded. First, the specification of the organizational structure is described. A template model is generated, which encapsulates the structure of the organization. On all existing levels of aggregation, the behavior of an organization can be described.

Taken together, this provides description of the behavior of an organization. In Section 3.2, will be explained how such dynamic behavior can be specified. In Section 3.3, the transition from template model to deployed model will be discussed.

3.1 Organization Structure

An organization structure is described by relationships between roles at the same and at adjoining aggregation levels and between parts of the conceptualized environment and roles. The specification of an organization structure uses the following elements:

(1) A role represents a subset of functionalities, performed by an organization, abstracted from specific agents who fulfill them.

Each role can be composed by several other roles, until the necessary detailed level of aggregation is achieved, where a role that is composed of (interacting) subroles, is called a composite role. Each role has an input and an output interface, which facilitate in the interaction (communication) with other roles. The interfaces are described in terms of interaction (input and output) ontologies: a vocabulary or a signature specified in order-sorted logic. At the highest aggregation level, the whole organization can be represented as one role. Such representation is useful both for specifying general organizational properties and further utilizing an organization as a component for more complex organizations. Graphically, a role is represented as an ellipse with white dots (the input interfaces) and black dots (the output interfaces). Roles and relations between them are specified using sorts and predicates from the structure ontology (see Table 1).

(2) An interaction link represents an information channel between two roles at the same aggregation level. Graphically, it is depicted as a solid arrow, which denotes the direction of possible information transfer.

(3) The conceptualized environment represents a special component of an organization model. Similarly to roles, the environment has input and output interfaces, which facilitate in the interaction with roles of an organization. The interfaces are conceptualized by the environment interaction (input and output) ontologies. These ontologies are defined using three types of predicates: to_be_observed, observation_result, and to_be_performed (see Table 1).

The internal specification for the environment can be conceptualized using one of the existing world ontologies (e.g., CYC, SUMO, TOVE). It can be defined by a set of objects with certain properties and states and with causal relations between objects. Graphically, the environment is depicted as a rectangle with rounded corners.

(4) An environment interaction link represents an information channel between a role and the conceptualized environment. Graphically, it is depicted as a dotted arrow, which denotes the direction of possible information transfer.

(5) An interlevel link connects a composite role with one of its subroles. It represents a transition between two adjacent aggregation levels. It may describe an ontology mapping for representing mechanisms of information abstraction. Graphically, it is depicted as a dashed arrow, which shows the direction of the interlevel transition.

Table 1. Ontology for formalizing organizational structure

Sort	Description
ROLE	Sort for a role
AGENT	Sort for an agent
ENVIRONMENT	Sort for the conceptualized environment
INTERACTION_LINK	Sort for an interaction link between two roles at the same aggregation level
INTERLEVEL_LINK	Sort for an interlevel link between two roles at two adjacent aggregation levels
ENVIRONMENT_INTERACTION_LINK	Sort for an environment interaction link between a role and the conceptualized environment
ONTOLOGY	Sort for an ontology
ONTO_MAPPING	Sort for an ontology mapping
STATE_PROPERTY	Sort for a state property expressed using some ontology
ACTION	Sort for an action performed in the environment
Predicate	**Description**
is_role: ROLE	Specifies a role in an organization
has_subrole: ROLE x ROLE	For a subrole of a composite role
source_of_interaction: ROLE x INTERACTION_LINK	Specifies a source role of an interaction
destination_of_interaction: ROLE x INTERACTION_LINK	Specifies a destination role of interaction
interlevel_connection_from: ROLE x INTERLEVEL_LINK	Identifies a source role of an interlevel link
interlevel_connection_to: ROLE x INTERLEVEL_LINK	Identifies a destination role of an interlevel link
initiator_env_interaction: ROLE x ENVIRONMENT_INTERACTION_LINK	Specifies a role-initiator in interaction with the environment
recipient_env_information: ROLE x ENVIRONMENT_INTERACTION_LINK	Identifies a role-recipient of information from the environment
part_of_env_in_interaction: ENVIRONMENT x ENVIRONMENT_INTERACTION_LINK	Identifies the conceptualized part of the environment involved in interaction with a role
has_input_ontology: ROLE x ONTOLOGY	Specifies an input ontology for a role
has_output_ontology: ROLE x ONTOLOGY	Specifies an output ontology for a role
has_input_ontology: ENVIRONMENT x ONTOLOGY	Specifies an input ontology for the environment
has_output_ontology: ENVIRONMENT x ONTOLOGY	Specifies an output ontology for the environment
has_interaction_ontology: ROLE x ONTOLOGY	Specifies an interaction ontology for a role
has_interaction_ontology: ENVIRONMENT x ONTOLOGY	Specifies an interaction ontology for the environment
has_onto_mapping: INTERACTION_LINK x ONTO_MAPPING	Identifies an ontology mapping
to_be_observed: STATE_PROPERTY	Describes a state property that will be observed in the environment
observation_result: STATE_PROPERTY x BOOLEAN_VALUE	Determines if a certain state property holds in the environment
to_be_performed: ACTION	Specifies an action that will be performed in the environment

3.2 Organizational Dynamics

At each aggregation level, it can be specified how the organization's behavior is assumed to be. To this end, organization dynamics are described by a dynamic representation, for each of the elements in an organization structure. The level of detail for specifying dynamics of an organization depends on its organizational type. Since the behavior of most mechanistic organizations is deterministic, dynamics for such organizations can only be modeled by a set of dynamic properties with high level of detail. In contrast, behavior of many organic organizations is defined loosely. Consequently, the dynamics of models for such organizations can be specified only partially; hence, actors (agents) can act autonomously.

The dynamics of each structural element are defined by the specification of a set of dynamic properties, formalized using the dynamic ontology (see Table 2).

Table 2. Dynamics ontology for formalizing properties of an organization

Sort	**Description**
DYNPROP	Sort for the name of a dynamic property
DPEXPR	Sort for the expression of a dynamic property
Predicate	**Description**
has_dynamic_property: ROLE x DYNPROP	Specifies a role dynamic property
has_dynamic_property: INTERACTION_LINK x DYNPROP	Identifies a dynamic property for an interaction link
has_dynamic_property: ENVIRONMENT x DYNPROP	Identifies a dynamic property for the conceptualized part of the environment
has_dynamic_property: ENVIRONMENT_INTERACTION_LINK x DYNPROP	Identifies a dynamic property for an environment interaction link
has_expression: DYNPROPx DPEXPR	Specifies an expression for a dynamic property

We define five types of dynamic properties:

(1) ***A role property*** **(RP)** describes the relationship between input and output states of a role, over time. The input and output states are represented as an assignment of truth-values to the set of ground atoms, expressed in terms of a role interaction (input or output) ontology.

For example, in the settings of a typical logistics company, a role property of a truck driver would be: if role Truck Driver receives a request from his manager to provide his coordinates, then role Truck Driver will generate this data for his manager.

(2) ***A transfer property*** **(TP)** describes the relationship of the output state of the source role of an interaction link to the input state of the destination role. Again, in the settings of a logistic company an example of a transfer company would be: if role Customer generates an order to role Transport Company, then Transport Company will receive this order.

(3) ***An interlevel link property*** **(ILP)** describes the relationship between the input or output state of a composite role and the input or output state of its subrole. Note that an interlevel link is considered to be instantaneous: it does not represent a temporal process, but gives a different view (using a different ontology) on the same information state. An interlevel transition is specified by an ontology mapping, which can include information abstraction.

(4) ***An environment property*** **(EP)** describes a temporal relationship between states or properties of objects of interest in the environment.

(5) ***An environment interaction property*** **(EIP)** describes a relation either between the output state of the environment and the input state of a role (or an agent) or between the output state of a role (or an agent) and the input state of the environment. On one hand, roles (or agents) are capable of observing states and properties of objects in the environment; on the other hand, they can act or react and, thus, affect the environment. We distinguish passive and active observation processes. For example, when some object is observable by a role (or an agent) and the role (or the agent) continuously keeps track of its state, changing its internal representation of the object if necessary, passive observation occurs. For passive observation, no initiative of the role or agent is needed. Active observation is always concerned with the role or agent's initiative.

3.3 Deployed Model and Agent Allocation

The generic or template model of an organization provides abstracted information concerning its structure and functioning. However, for a more detailed analysis, a deployed model is needed. It should be based on both unfolded generic relations between roles, as defined in the template model, and on creating new role instances. Moreover, the environment (or scenario) influences the specification of a deployed model considerably. Subsequently, different deployed models can be specified for different scenarios, using the same template model of an organization. For formalizing the structure and behavior of a deployed model, the same ontologies are used as for formalizing a template model.

The deployed model abstracts from the actual agent allocation but provides the detailed specifications for the behavior of role instances. Based on these specifications, a set of requirements is formulated for each role instance. These requirements (restricting and defining behavior) are imposed onto the agents, who will eventually enact these roles. The formalization of the allocation principles is performed in line with the formalization of the template and the deployed models, using the predicate allocate_to. In some scenarios, a complex role can act as a single aggregated role and, thus, representing its constituting subroles. In such cases, an agent(s) can be assigned to the complex role. If, for some reason, an allocated agent is not anymore capable of enacting a certain role, dynamic reallocation of another agent will take place, as described in Section 6.

4 Formal Specification of the Organization Model

In the previous section, the elements of the methodology were introduced. The current section provides the formal specification of them.

4.1 Structural Properties

Structural properties represent relations between structural elements of the organization. They are specified in a sorted first-order predicate logic, based on the structure ontology. For example, in the settings of a logistics company, subroles Fleet Manager (FM) and Load Manager (LM) belong to the same composite role Operational department (OP). Formally:

has_subrole(OP, FM) ∧ has_subrole(OP, LM)

Often, structural properties are valid during the whole period of organization existence and can be considered as static. But in rapidly developing and adapting organizations, structural change processes gain special importance. Structural properties for such organizations get a temporal dimension and can be considered as a subclass of dynamic properties.

4.2 State and Dynamic Properties

A dynamic property represents a relation in time either between (input or output) states of roles or a (input or output) state of a role and a (input or output) state of the environment. To achieve this, every role as well as the conceptualized part of the environment has assigned state ontologies for input and output states. A state for ontology Ont is an assignment of truth-values to the set At(Ont) of ground atoms expressed in terms of Ont. The set of all possible states for state ontology Ont is denoted by STATES(Ont). A state property is defined by a formula over a state ontology.

Role or environment states are related to state properties via the formally defined satisfaction relation |=, comparable to the Holds-predicate in situation calculus: state(γ, t, output(r)) |= p, which denotes that state property p holds in trace γ at time t in the output state of role r.

For a fixed, linearly ordered, time frame TIME (e.g., natural or real numbers), a trace γ over a state ontology Ont is defined as a mapping γ: TIME → STATES(Ont) or, in other words, a sequence of states γ_t (t ∈ TIME) in STATES(Ont). The set of all traces over state ontology Ont is denoted by TRACES(Ont).

Dynamic properties (e.g., for roles, environment, and links) are specified in the language Temporal Trace Language (TTL) [12], based on a sorted first-order predicate logic with sorts TIME for time points and TRACE for traces, using quantifiers over time and logical connectives. The specification of properties in TTL is supported by a dedicated editor [13]. For examples of dynamic properties in formal form, see Sections 6.

5 Verification and Validation

The model as introduced in this paper offers the means for both role-centered and agent-centered verification and validation. This section briefly discusses these.

Role-centered verification techniques may be used for analysis of both template and deployed models of organizations. Subsequently, inconsistencies and bottlenecks in an

organization can be detected. Agent-centered verification techniques are used for analyzing scenarios with roles of an organization model, allocated to (human) agents.

For those cases where empirical traces (i.e., sequences of states) of the processes within an organization are (partially) available, it is possible to validate properties against such a trace. For example, a trace can be obtained from log-files of a company. If an empirical trace is given informally, the first step is to formalize it (by hand), using formal state ontologies. If it is already given in a formal form, the first step is to translate (e.g., automatically) the formal representation into one based on ontologies used in the organization model. For the trace that has been generated by simulation, translation into the right formal format can be automated as an interface between the simulation environment and the checking environment. Once such a trace is in the right formal form, it is possible to verify dynamic properties of the organization (including structural properties), using dedicated checking software.

As input for the verification software, a formalized trace and a formalized property have to be provided. Given such input, after automatic verification of the given property against the given trace, the software will generate a decision (positive or negative). The positive decision denotes that the property holds with respect to the given trace. In case of a negative decision, the software explains why the property does not hold. This type of verification is shown in the case study in Section 6.

Another verification method uses a simulation model based on the specification of the dynamic properties of the lower aggregation level for checking the properties of the higher aggregation level. This verification method is supported by the dedicated checking software [13].

When an organization model is specified including dynamic properties at different aggregation levels, it is not automatically guaranteed that these properties at different levels fit to each other. A verification process that relates properties at one aggregation level to those of another level (e.g., as in compositional verification) can reveal incompleteness or inconsistencies. In the case study presented in the next section, it is shown how such a mutual verification process can be performed.

6 Case study

In this section, a case study is described. In parallel, the newly developed graphical representation of organization models is introduced. This case study was done within the project DEAL (Distributed Engine for Advanced Logistics). For the project description, we refer to *http://www.almende.com/deal/*. A template organizational model was created, based on the informal description of the structure and functioning of the large Dutch logistics company. To secure anonymity of the company, the real names of the organizational units were substituted by general ones,.

At the highest level (abstraction level 0) we represent the whole organization as one role. At abstraction level 1, the organization consists of two interacting roles: TC and CI (see Fig.1). Note, that the organizational model is depicted in a modular way; i.e., components of every aggregation level can be visualized and analyzed both separately and in relation to each other. Consequently, scalability of graphical representation of an organizational model is achieved.

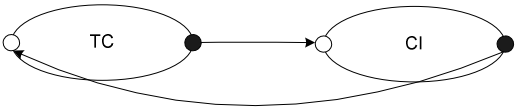

Fig. 1. Representation of the organization at abstraction level 1, which consists of role Transport Company (TC) and role Customer Interaction (CI)

At abstraction level 2 role TC can be refined into three interacting roles: ST, CR, and OP (see Fig.2). All interactions with a customer are conducted within CI role. At abstraction level 2 it consists of two roles: TCR and C (see Fig. 2). Role TCR produces at its output messages from CR and ST departments of the transport company, i.e., CR and ST roles stand as company representatives in certain interactions with a customer. Therefore, the input state of role TCR has influence on the output state of role CR and vice versa. The same holds for role ST.

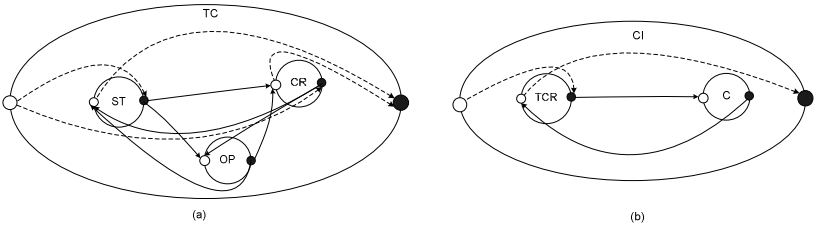

Fig. 2. Representation of (a) the Transport Company (TC) and (b) the Customer Interaction role (CI) at abstraction level 2

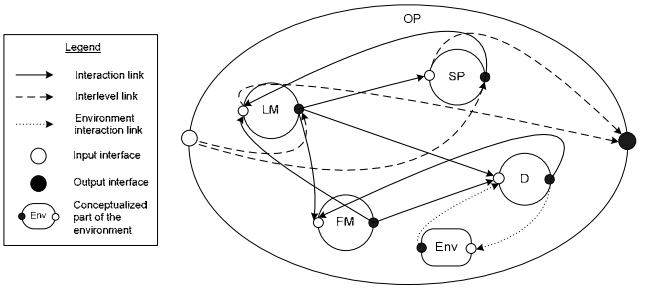

Fig. 3. Representation of the operational department at abstraction level 3

The corresponding dynamic properties may be specified at abstraction level 0 and can be further refined into basic properties at lower abstraction levels. In our case study, we particularly concentrate on the structure and functioning of the OP (see Fig. 3), part of the TC.

Table 3. Role names, abbreviations, and descriptions for the organizational model in the case study

Role name	Abbreviation	Description
Transport Company	TC	Provides logistic services to customers
Customer Interaction	CI	Identifies interaction rules between a customer and the transport company
Strategy and Tactical Department	ST	Performs analysis and planning of company activities; considers complaints from customers; analyses the satisfaction level of a customer by means of surveys and questionnaires
Custom Relations Department	CR	Handles requests from customers
Operational Department	OP	Responsible for direct fulfillment of the order from a customer
Transport Company Representative	TCR	Mediator role between a customer and the transport company
Customer	C	Generates an order for the transport company; sends inquiries about the delivery status
Sales Person	SP	Assigns an order to a certain load manager, based on the type and the region of a delivery
Load Manager	LM	Assigns orders to suitable trucks and available drivers; assigns fleet managers to drivers; provides CR department with up-to-date information about delivery; provides a driver with instructions in case of a severe problem; informs CR department about possible delays with delivery
Fleet Manager	FM	Keeps constant contact with the assigned drivers; updates automatic support system with actual data on the delivery status; provides consultations for drivers in case of minor problems in transit
Driver	D	Delivers goods; informs a superior fleet manager about the delivery status; interacts (by means of observations and actions) with the conceptualized part of the environment
Environment	Env	Represents the conceptualized environment; in this case study only a driver interacts with it

The static aspects of the considered organization have been formally described in the organization structure specification. The sets of dynamic properties for the components of the organization structure have been identified at different abstraction levels. For example, consider the information distribution property of role OP called RP1(OP), specified at abstraction level 2. Informally, when a severe problem with some delivery occurs, OP should generate a message to CR about possible delay. Formally specified:

$\forall\gamma$:TRACE $\forall$t1:TIME $\exists$T:TRUCK_TYPE $\exists$D:DRIVER $\exists$ON:ORDER_NUM state(γ, t1, environment))|= truck_state(T, incident, severe_incident) $\wedge$ truck_property(T, operated_by, D) $\wedge$ order_property(ON, assigned_to, D)$\Rightarrow$ $\exists$t2:TIME t2>t1 state(γ, t2, output(OP))|=communicate_from_to(OP, CR, inform, order_state(ON, delay, severe_incident)),

where Table 4 provides the description of the predicates.

Table 4. Predicates for formalizing the dynamic properties used in the examples

Predicate	Description
communication_from_to(r1:ROLE, r2:ROLE, s_act:SPEECH_ACT, message:STRING)	Specifies the speech act *s_act* (e.g., inform, request, ask) from role-source *r1* to role-destination *r2* with the content *message*
deliverable_object(on: ORDER_NUM, desc:STRING)	Assigns the order number *on* with the description *desc* to the object that has to be delivered
truck_property(trt:TRUCK_TYPE, operated_by, d:DRIVER)	Assigns the driver *d* to a truck of the type *trt*
order_property(on:ORDER_NUM, assigned_to, d:DRIVER)	Assignes the order *on* to the driver *d*
order_property(on:ORDER_NUM, deadline, d_value:INTEGER)	Identifies the deadline *d_value* for the order *on*
truck_state(trt:TRUCK_TYPE, st:STATE, descr:STATE_DESCRIPTION)	Denotes the state *st* with the state description *descr* of a truck of the type *trt*
order_state(on:ORDER_NUM, st:STATE, descr:STATE_DESCRIPTION)	Specifies the state *st* with the state description *descr* of the order with the number *on*

This property can be logically related to the conjunction of dynamic properties at a lower abstraction level 3 in the following way:

$$EP1(Env, T, severe\_incident) \wedge EIP1(Env, D) \wedge RP1(D) \wedge TP1(D, FM) \wedge RP2(FM) \wedge TP2(FM, LM) \wedge RP3(LM) \wedge ILP1(LM, OP) \Rightarrow RP1(OP)$$

Using the verification technique, as described in Section 5, can be shown that the latter logical relation indeed holds. Between brackets, the abbreviations of the dynamic properties, are provided, conform the specification provided in Section 3.2. In the environment occurs an event: a severe incident with the truck T, for which role D is responsible (EP1). D observes this incident (EIP1) and reacts by generating a request for advice to FM (RP1). FM receives this request (TP1). FM is not empowered of making decisions in such situations; therefore s/he propagates the request further to LM (RP2). LM receives the request (TP2). LM officially identifies the incident as severe (RP3) and outputs the notification about a possible delay from role OP to CR (ILP1). Thus, by a manually conducted, mathematical proof, the previously identified logical relation between two adjoining aggregation level spaces indeed holds. In general, attempting to set up such a manually conducted, mathematical proof can reveal missing premises or other shortcomings such as inconsistencies.

In the deployed model for the considered case study, all roles specified at abstraction levels 1 and 2 have one-to-one mapping to the role instances. While roles LM, FM, and D (defined at abstraction level 3) have multiple instances; e.g., LM and FM are represented differently in different geographical regions and, subsequently, different types of trucks and professional skills of drivers are required for different kinds of deliveries. The deployed model for our case study (see Fig. 4) is based on the assignment relation. For example, assigned_to(D2, FM1) denotes that a middle-size truck and his driver are assigned to the fleet manager in eastern Europe and the region belonging relation in_region(D1, LM1) specifies that both a big-size truck driver and a load manager should belong to the same region in eastern Europe).

When a template and a deployed organizational model are specified, agent allocation principles should be formulated. The capabilities of agents, essential for this case study were identified. For example, a driver-agent can drive a truck; hence, he has a driver license of a certain type, has acceptable results of medical tests etc. In addition, the allocation requirements for role instances were formulated; e.g., in order to enact role LM, an agent should have working experience as a senior manager in logistics for at least 3 years.

Let us briefly consider the scenario reconstructed from empirical data of the transport company, using specified organizational model:

(1) A Customer places an order by means of a contact with TCR (CR department in this case) in CI.
(2) Inside TC this order is being transmitted from CR to OP.
(3) Within OP the order is distributed by SP to LM1.
(4) LM1 assigns the order to D1, D1 is associated with FM1 (see Fig. 4).
(5) D1 starts delivery, then after some time a severe incident occurs with his truck.
(6) D1 asks for help FM1, who incapable of making a decision in this case.
(7) FM asks for a solution LM1, who decides to send another truck to proceed with delivery.
(8) Now D1 is reallocated to another truck and driver, who picks up goods and continues delivery.
(9) At the same time LM1 informs CR about possible delay with delivery.
(10) CR, who shares the same knowledge with TCR, informs the Customer about possible delay.
(11) D1 successfully finishes delivery and the Customer is being informed about that.

Using formal state ontologies (see Tables 1 and 2), we formalized this trace in the LEADSTO environment [13]. A formalized empirical trace is useful for analysis of organizational functioning. For the case study, we identified several properties of interest that can be automatically verified against the trace. Let us consider some of these properties.

(1) Delivery successfulness
Informally: the order has been fulfilled. Formally:

$\exists t{:}\mathrm{TIME}\ \exists O{:}\mathrm{ORDER\_NUM}\ \mathrm{state}(\gamma, t, \mathrm{environment}) \models \mathrm{order\_state}(O, \mathrm{delivered}, \mathrm{final\_report})$

An automatic verification, as mentioned in Section 5, confirmed that this property holds against the formalized empirical trace.
(2) Customer notification
Informally: always if a severe problem occurs with the truck and the driver, who was fulfilling the order of some customer, then this customer should be notified about possible delay with delivery. Formally:

$\forall \gamma{:}\mathrm{TRACE}\ \forall t1{:}\mathrm{TIME}\ \exists T{:}\mathrm{TRUCK\_TYPE}\ \exists D{:}\mathrm{DRIVER}\ \exists ON{:}\mathrm{ORDER\_NUM}\ \mathrm{state}(\gamma, t1, \mathrm{environment})) \models \mathrm{truck\_state}(T, \mathrm{incident}, \mathrm{severe\_incident}) \wedge \mathrm{truck\_property}(T, \mathrm{operated\_by}, D) \wedge \mathrm{order\_property}(ON, \mathrm{assigned\_to}, D) \Rightarrow \exists t2{:}\mathrm{TIME}\ t2 > t1\ \exists TCR{:}\mathrm{ROLE}\ \mathrm{state}(\gamma, t2, \mathrm{input}(\mathrm{customer})) = \mathrm{communicate\_from\_to}(TCR, \mathrm{customer}, \mathrm{inform}, \mathrm{order\_state}(ON, \mathrm{delay}, \mathrm{customer\_report}))$

Again automatic verification confirmed that this property holds against the trace.
(3) Delivery accuracy
Informally: the order has been fulfilled on time. Formally:

$\exists t{:}\mathrm{TIME}\ \exists O{:}\mathrm{ORDER\_NUM}\ \exists d\_value{:}\mathrm{integer}\ \mathrm{state}(\gamma, t, \mathrm{environment}) \models \mathrm{order\_state}(O, \mathrm{delivered}, \mathrm{final\_report}) \wedge \mathrm{order\_details}(O, \mathrm{deadline}, d\_value) \wedge d\_value \geq t$

This property does not hold with respect to the trace. The next logical step in analysis of the causes for property failing would be to check if some incident occurred in transit. In case that a severe incident happened with the truck and the agent (a truck driver) was incapable of performing his role any more, the next step would be to verify whether or not enough time is available for a role reallocation. Subsequently, analysis of organization functioning can be continued until all inquiries about delivery are satisfied.

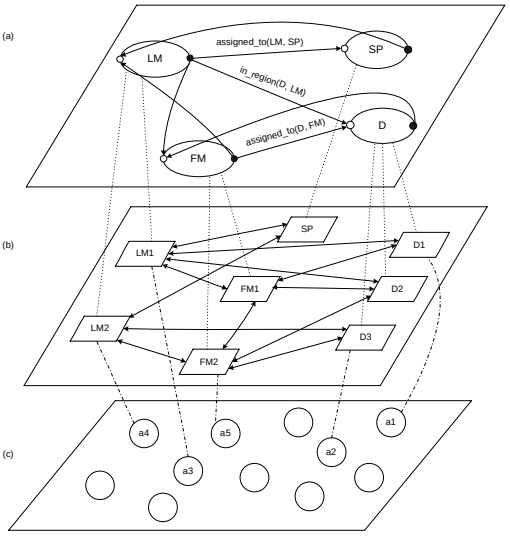

Fig. 4. The operational department of the transport company represented at abstraction level 3, with (a) the template model (b) the deployed model, and (c) agent allocation

7 Discussion

Both in human society and for software agents, organizational structure provides the means to make complex, composite dynamics manageable. To understand and formalize how exactly organization structure constrains composite dynamics is a fundamental challenge in the area of organizational modeling. The modeling approach presented in this paper addresses this challenge. It concerns a method for formal specification of organizations, which can capture both structural and dynamic aspects of organizations and provides the means for (i) representation of organization structure, (ii) simulations of different scenarios, (iii) analysis of organization, verifying static and dynamic properties against (formalized) empirical data or simulated scenarios, (iv) diagnosis of inconsistencies, redundancies, and errors in structure and functioning. Additionally, the environment is integrated as a special component within the organization model.

Specification of organization structure usually takes the form of pictorial descriptions, in a graph-like framework. These descriptions often abstract from detailed dynamics within an organization. Specification of the dynamic properties of organizations, on the other hand, usually takes place in a completely different conceptual framework; these dynamic properties are often specified in the form of a set of logical formulae in some temporal language. The logical relationships express the kind of relations between dynamics of parts of an organization, their interaction, and dynamic properties of the organization as a whole, which were indicated as crucial by Lomi and Larsen [14] in their introduction.

This paper shows how pictorial descriptions, in a graph-like framework, and a set of logical formulae in some temporal language can be combined in one agent-based

modeling approach. Inspection can be done on the abstraction level preferred and both the pictorial and formal specifications of the dynamic properties can be inspected. Five essential types of dynamic properties characterizing behavior of main structural components of an organization model (including environment) are identified. So far, more complex cases of organizational behavior (e.g., the synchronization problem for joint action) were not discussed. For example, in the case of joint lifting by roles or agents more sophisticated types of dynamic properties are needed; e.g., combined role properties that define temporal relations between a number of states for some set of roles and a number of states of another set of roles. Furthermore, the approach proposed here supports formal specification and verification for both static and dynamic properties. This possibility is especially useful for diagnosis of inconsistencies, redundancies, and errors in structure and functioning of real organizations and providing recommendations for their improvement (e.g., by way of evaluating of performance indicators).

Compared to most organization-oriented, multi-agent system, design approaches [1, 4, 5, 22], our model allows any number of aggregation levels in the organization model, which makes it more suitable for modeling and analyzing real organizations. While a role aggregation relation is considered to be crucial for representing an organizational model, other types of relations between roles should also be taken into account. For example, a role specified in a template model and its corresponding role instances defined in a deployed model are related by means of a generalization relation. Furthermore, it would be interesting to investigate how role hierarchies, based on generalization, and other types of relations can be used in the specification of a template model.

Let us now consider a case in which agents show autonomous behavior, independent of (or sometimes conflicting to) organizational rules and goals. In this case, an agent behavior can be specified from the positions of sociological theories, which take into account an individual behavior of social actors. One of such theories, the Sociology of Organized Action studies an organization functioning beyond its formal rules and is used for specifying informal coordination mechanisms in agent organizations [21]. To tackle the forthcoming compatibility problems from the relationships between formally predefined organizational model and agent autonomous behavior, further investigation will be undertaken. When the latter would be accomplished, many types of modern organizations could be modeled.

In the case of highly dynamic organizations (e.g., self-organizing and organic organizations), organizational change is a crucial and frequent process. Due to their high complexity, such organizations are difficult to investigate. However, different simulation techniques can help in providing further insights into mechanisms of functioning of such organizations. For the latter purpose, research has been conducted based on the introduced formal model [8]. In [6] a simulation technique is suggested that can be used for evaluation of different alternatives of an organizational structure with respect to the task performance and reorganization when necessary.

In addition, the different types of modern organizations should be taken in consideration, as organization theory [15, 20] classifies and describes them. It would be useful to develop and formally specify the templates capturing essential structural and dynamic aspects of most frequently encountered types of organizations. Such templates would be of great help for organization designers and analysts.

In conclusion, this paper introduced a new, formal, fully traceable method on modeling and analyzing (multi-agent) organizations. It comprises both static and dynamic aspects as well as environment representation. Hence, it provides the basis of a formal framework, which provides the means for both the design and for the automatic validation and verification of organizations.

Acknowledgments

This research was partially supported by the Netherlands Organization for Scientific Research (NWO) under project number 612.062.006. SenterNovem is gratefully acknowledged for funding the projects Cybernetic Incident Management (CIM) and Distributed Engine for Advanced Logistics (DEAL) that also funded this research partially. Further, we thank the reviewers for their detailed comments on the original manuscript.

References

1. Amiguet, M., Müller, J.P., B'aez-Barranco, J.A., Nagy, A.: The MOCA platform. Lecture Notes in Computer Science (Multi-Agent-Based Simulation) 2581 (2003) 70-88
2. Bresciani, P., Giorgini, P., Giunchiglia, F., Mylopoulos, J., Perini, A.: Tropos: An agent-oriented software development methodology. Autonomous Agents and Multi-Agent Sytems 8 (2004) 203-236
3. Dastani, M., Hulstijn, J., Dignum, F., Meyer, J.J.Ch.: Issues in multiagent system development. In Jennings, N.R., Sierra, C., Sonenberg, L., Tambe, M., eds.: Proceedings of the third International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2004). Volume 2, IEEE Computer Society (2004) 922-929
4. Esteva, M., de la Cruz, D., Sierra, C.: ISLANDER: An electronic institutions editor. In Gini, M., Ishida, T., Castelfranchi, C., Johnson, W.L., eds.: The First International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2002). Volume 3, ACM (2002) 1045-1052
5. Ferber, J., Gutknecht, O., Michel, F.: From agents to organizations: An organizational view of multi-agent systems. Lecture Notes in Computer Science (AOSE) 2935 (2003) 214-230
6. Furtado, V., Melo, A., Dignum, V., Dignum, F., Sonenberg, L.: Exploring congruence between organizational structure and task performance: A simulation approach. Lecture Notes in Computer Science [in the same volume]
7. Hannoun, M., Sichman, J.S., Boissier, O., Sayettat, C.: Dependence relations between roles in a multi-agent system: Towards the detection of inconsistencies in organization. Lecture Notes in Computer Science (Multi-Agent-Based Simulation) 1534 (1998) 169-182
8. Hoogendoorn, M., Jonker, C.M., Schut,M.C., Treur, J.: Modelling the organisation of organizational change. In Giorgini, P., Winikoff, M., eds.: Proceedings of the Sixth International Workshop on Agent-Oriented Information Systems (AOIS'04). (2004) 26-46
9. Horling, B., Lesser, V.: A survey of multi-agent organizational paradigms. The Knowledge Engineering Review 9 (2005) 281-316
10. Hübner, J.F., Sichman, J.S., Boissier, O.: A model for the structural, functional, and deontic specification of organizations in multiagent systems. Lecture Notes in Computer Science (Advances in Artificial Intelligence) 2507 (2002) 118-128

11. Hübner, J.F., Sichman, J.S., Boissier, O.: S-MOISE+: A Middleware for developing Organized Multi-Agent Systems. Lecture Notes in Computer Science [in the same volume]
12. Jonker, C.M., Treur, J.: A temporal-interactivist perspective on the dynamics of mental states. Cognitive Systems Research Journal 4 (2003) 137-155
13. Jonker, C.M., Treur J., Wijngaards W.C.A.: A temporal-modelling environment for internally grounded beliefs, desires, and intentions. Cognitive Systems Research Journal 4(3) (2003) 191-210
14. Lomi, A., Larsen, E.R. (eds.): Dynamics of Organizations: Computational Modeling and Organization Theories. Cambridge, MA: The M.I.T. Press (2001)
15. Mintzberg, H.: The Structuring of Organizations. Prentice Hall, Englewood Cliffs (1979)
16. Miles, R.E., Snow, C.C., Mathews, J.A., Miles, G., and Coleman, H.J.: Organizing in the Knowledge Age: Anticipating the Cellular Form. Academy of Management Executive 11 (1997) 7-20
17. Morgan, G.: Images of organizations. SAGE Publications, Thousand Oaks London New Delhi (1996)
18. Omicini, A.: SODA: Societies and infrastructures in the analysis and design of agent-based systems. Lecture Notes in Computer Science (AOSE) 1957 (2001) 185-193
19. Prietula, M., Gasser, L., Carley, K.: Simulating Organizations. MIT Press (1997)
20. Scott, W.R.: Institutions and organizations. 2nd ed. SAGE Publications, Thousand Oaks London New Delhi (2001)
21. Sibertin-Blanc, C., Amblard, F., Mailliard, M.: A coordination framework based on the Sociology of the Organized Action. Lecture Notes in Computer Science [in the same volume]
22. Zambonelli, F., Jennings, N. R., Wooldridge, M.: Developing multiagent systems: the Gaia Methodology. ACM Transactions on Software Engineering and Methodology (TOSEM) 12(3) (2003) 317-370