A Formal Framework for Organization Modeling and Analysis

Viara Popova[1] and Alexei Sharpanskykh[2]
[1] De Montfort University, UK
[2] Vrije Universiteit Amsterdam, The Netherlands

Abstract. This chapter introduces a formal framework for modeling and analysis of organizations. It allows representing and reasoning about all important aspects of artificial and human organizations structured in a number of views, including performance-oriented, process-oriented, power- and interaction-related aspects. The framework provides means to model formal (pre)defined organizational structures and dynamics, informal relations and behavior of organizational actors. The meaning attached to the modeling concepts is specified based on the literature from Social Science. Unlike many existing organization modeling approaches the proposed framework has formal foundations based on the order-sorted predicate logic which enables different types of analysis of organizational specifications of particular views and across views. The framework allows scalability of modeling and analysis of complex organizations, by considering them at different aggregation levels. Furthermore, the framework provides support for real-time management of organizational processes. The framework was applied in several case studies, one of which is discussed here.

INTRODUCTION

The modern world is unthinkable without organizations. The rapid scientific, societal and technological development of the last centuries, coupled with the changed environmental conditions gave rise to a great diversity of organizational forms and types of interaction between them. The structural and behavioral complexity of organizations is interdependent on the complexity of the environment, in which these organizations are situated. The complex, dynamically changing environment with insufficient resources often creates challenging obstacles for the satisfaction of the primary goals of any organization – to survive and prosper. To be successful, an organization should effectively and efficiently organize its internal structure and activities so that the fit with the environment is achieved. In reality these requirements are difficult to fulfill, since no universally applicable recipes exist that ensure the successfulness of an organization at all times and all cases. Therefore, most modern organizations suffer from various performance inefficiencies and inconsistencies that may have consequences for the organizational vitality. Often only a small number of these flaws can easily be identified, but can be revealed using more profound analysis methods.

Many of the techniques for analysis of organizational performance developed in organization theory are informal and imprecise, which undermines the feasibility and rigor of the analysis. For more precise evaluation of the organizational performance, for identification of performance bottlenecks and conflicts, detailed organizational analysis based on a formal organization model should be performed. Furthermore, a formal organization model constitutes a basis for many

automated processes within enterprises (e.g., computer integrated manufacturing) and provides a foundation for inter-enterprise cooperation.

To enable formal analysis, this Chapter introduces a formal modeling framework that allows representing diverse aspects of organizational reality, within several perspectives, e.g. the process-related, performance-related and organization-related perspectives. Since individuals often exert significant influence on the organizational dynamics, also aspects related to human behavior are considered explicitly. The characteristics of the framework include:

- it allows the representation and analysis of organization models at different levels of abstraction in order to handle complexity and increase scalability;
- it enables formal verification and validation of models of different perspectives on organizations;
- it enables simulation for experimenting and testing hypotheses on the organizational behavior under different circumstances;
- it proposes computational analysis methods across multiple perspectives on organizations;
- it supports and controls the execution of organizational scenarios and the evaluation of organizational performance.

The framework proposes a wide spectrum of means for modeling and analysis of structures and dynamics of organizations of different types including mechanistic organizations that represent systems of hierarchically linked job positions with clear responsibilities and organic organizations characterized by highly dynamic, constantly changing, structure with non-linear behavior. Although the structure and behavioral rules for organic organizations can hardly be identified and formalized, by performing agent-based simulations with changing characteristics of proactive agents, useful insights into the functioning of such organizations can be gained. Furthermore, the framework supports reuse of parts of models.

The Chapter is organized as follows. First the related literature is presented. Then the formal foundations of the framework are described. After that the case study used for illustration is introduced. An overview of the four modeling views is given. It is then discussed how the framework can be used in practice. Finally, the methods for organizational analysis are described. The Chapter ends with conclusions and future research directions.

RELATED LITERATURE

This Section provides a general overview of organization modeling and analysis approaches and techniques developed in three areas: organization theory, enterprise information systems and organization-oriented multi-agent systems.

Organization theory

Organization theory distinguishes three aggregation levels, at which processes and structures of human organizations are studied: *micro*, *meso*, and *macro*. At the individual (*micro*) level the behavior of individuals and work groups is investigated. At the level of the whole organization (*meso* level) different aspects of the organizational structure and dynamics are considered. At the global (*macro*) level the interaction between the organization and its environment including other organizations, society, markets, is considered.

The specifications of organizations normally used in Organization Theory are represented by informal or semi-formal graphical descriptions that illustrate aspects of organizations at some aggregation level (Mintzberg, 1979) (e.g., decision making, authority relations). Often they are specified in an imprecise and ambiguous way, making it difficult to apply them in practice. One of the attempts to identify concrete, practically applicable recommendations for designing

organizations is within contingency theory (Burton & Obel, 2004). The key thesis is that the structure and behavior of an organization should be defined based on particular environmental characteristics. To support this, the contingency theory identifies a number of generic principles for designing effective organizations which should be carefully adapted in the context of this particular organization. In the adaptation process inconsistencies and inefficiencies may be introduced that cannot be foreseen and identified by the contingency theory. To identify these inconsistencies and inefficiencies analysis techniques are required.

Another class of approaches for specifying quantitative organization models with precise semantics has been proposed in the System Dynamics Theory. Organizational descriptions specified in System Dynamics are based on numerical variables and equations that describe how these variables change over time. Such specifications can be computationally effective; however they lack ontological expressivity to conceptualize relations of different types of organizations. Furthermore, they abstract from single events, entities and actors and take an aggregate view on the organizational dynamics. Therefore, such approaches cannot be used for modeling organizations at the micro level.

The problem of the limited ontological expressivity of modeling has been addressed in the area of enterprise information systems, considered in the following Section.

Enterprise information systems

An enterprise information system (EIS) is any computing system automating the execution of process(es) of an enterprise. EISs are often built based on enterprise architectures. An *enterprise architecture* (EA) is an enterprise-wide, integrating framework used to represent and to manage enterprise processes, information systems and personnel, so that key goals of the enterprise are satisfied. Many different EAs have been developed CIMOSA (Bernus, Nemes & Schmidt, 1998), TOVE (Fox *et al*, 1997), ARIS (Bernus, Nemes & Schmidt, 1998). Based on common features of these architectures a generalized meta-framework GERAM (Generalized Enterprise Reference Architecture and Methodology) was developed (Bernus, Nemes & Schmidt, 1998). GERAM provides a template for the development and comparison of enterprise modeling frameworks. GERAM describes a number of dedicated views on enterprises. *The function view* concerns structural and behavioral aspects of the business processes of an enterprise. The following techniques are used: IDEF standards (Bernus, Nemes & Schmidt, 1998), statecharts, Petri-nets (Bernus, Nemes & Schmidt, 1998), semi-formal languages (BPML, etc.). *The information view* describes knowledge about objects as they are used and produced. The following data models are used: Entity-Relationship-diagrams, object-oriented representations, UML class diagrams. *The resource view* considers resources of an enterprise often modeled as separate entities in the existing frameworks with varying level of detail. *The organization view* defines responsibilities and authorities on processes, information and resources.

Although many architectures include a rich ontological basis for modeling different views, most of them provide limited support for automated analysis of models, addressed in the category *Enterprise Engineering Tools* of GERAM, primarily due to lack of formal foundations in these architectures.

Within several frameworks, analysis methods for particular views have been developed (e.g., process-oriented modeling techniques for the function view (van der Aalst *et al*, 2003), ABC-based techniques for the performance-oriented view (Tham, 1999)). Much less attention has been devoted to analysis performed across different views that allows investigating a combined influence of factors from different views on the organizational behavior. In (Dalal *et al*, 2004) an integrated framework for process and performance modeling is described that incorporates accounting/business parameters into a formal process modeling approach based on Petri-nets. However, key aspects as authority relations, goals, individual behavior are not considered. Another formal framework for business process modeling is described in (Koubarakis &

Plexousakis, 2002) focusing on formal goal-oriented modeling using situation calculus. Modeling and analysis of processes and other organizational concepts are not properly addressed. A formal framework for verifying models specified in Unified Enterprise Modeling Language (UEML) is proposed in (Chapurlat, Kamsu-Foguem & Prunet, 2006). It identifies a general idea to use conceptual graphs for verifying enterprise models; however, neither technical nor experimental results are provided.

Usually EISs are based on predefined organizational specifications that guide and/or control processes performed by organizational actors. To enable modeling and analysis of behavior of organizational actors in different organizational and environmental settings, the agent paradigm is particularly useful.

Organization-oriented multi-agent systems

An agent is a piece of software with the ability to perceive its environment (virtual or physical), reason about its perception of this environment and act upon the environment. Interactions among agents often take place in the context of certain organizational formations (structures). Such structures may be intentionally designed to enforce rules on agent behavior or emerge from the non-random and repeated patterns of interactions among agents. The organizational structure provides means to coordinate the execution of tasks in a multi-agent system (MAS) and to ensure the achievement of organizational goals.

In (Horling & Lesser, 2005) several types of organizational structures are distinguished including hierarchies, holarchies, coalitions, teams, congregations, and federations, providing the agents with different degrees of autonomy. Usually the behavior of agents is restricted by a set of norms defined at different aggregation levels of the organizational structure. Currently many approaches for modeling normative MASs have been proposed in the literature. Often organizational structures are specified in terms of roles defined as abstract representations of sets of functionalities performed by an organization. Also, in the approach proposed in the paper diverse organizational structures are modeled as roles.

In Chapter 3 "Modelling Dimensions for Agent Organizations" by L. Coutinho, O. Boissier and J. Sichman eight dimensions for modeling of agent organizations are introduced: structures of roles and groups, dialogical interaction structures, goal/task decomposition structures, normative structures, environment, organizational evolution, organizational evaluation and ontologies. The modeling framework proposed in this chapter considers explicitly all these dimensions, except for the organizational evolution. Nevertheless, this dimension can be also modeled by dynamics modeling means of the proposed framework.

The GAIA methodology (Zambonelli, Jennings & Wooldridge, 2003) addresses two development phases: analysis and design. At the analysis phase, roles and relations between them are identified. During design societies of agents are specified. GAIA does not capture the internal aspects of agents. The interaction of agents with the environment is not treated separately. In contrast, the approach proposed in this chapter considers the internals of agents and interaction with the environment explicitly.

The original AGR organizational model (Ferber & Gutknecht, 1998) considers only structural aspects of organizations. Each organizational AGR model comprises a set of interrelated groups consisting of roles. AGR enforces no constraints on the internal architecture of agents. In Chapter 4 "From AGR to MASQ: towards an integral approach of organizations, environment and institutions in multi-agent systems" by J. Ferber, T. Stratulat, and J. Tranier an extension of the AGR model is proposed - the meta-model MASQ (Multi-Agent System based on Quadrants). This model is based on a 4-quadrant framework, where the system analysis and design is performed along two dimensions: an interior/exterior dimension and an individual/collective dimension. The framework proposed in this chapter addresses all four quadrants, including mental states of agents and their externally observable behavior, physical world and its

components, social norms and interaction conventions, shared knowledge. Furthermore, the framework of this chapter provides a more refined view on each quadrant by distinguishing specific types of concepts, states, and relations, which will be introduced further in this chapter.

MOISE (Hannoun *et al*, 2000) is an organizational model that provides descriptions along three levels: the individual level of agents; the aggregate level of large agent structures; the society level of global structuring and interconnection between agents and structures. In (Hubner, Sichman & Boissier, 2002) the methodology is extended with functional aspects (such as tasks, plans, and constraints on the behavior of a MAS). Our framework provides a more elaborated view on the functional aspects.

The TROPOS methodology (Bresciani, 2004) addresses three development phases of MASs: analysis, design and implementation. During analysis, a list of functional and non-functional requirements for the system is identified. During design, the structure and behavior of a system in terms of its subsystems related through data, control and other dependencies are defined. The implementation phase maps the models from the design phase into software by means of Jack Intelligent Agents. While TROPOS aims mostly at designing and implementing robust and orderly multi-agent systems, the framework proposed in this chapter can be used for modeling and analysis of both artificial organizations of agents and human organizations. Thus, many aspects and relations inherent in human organizations are not used in TROPOS (e.g., power relations, workflow modeling).

The OperA framework (Dignum, 2003) focuses on social norms and explicitly defines control policies to establish and reinforce these norms. The framework comprises three components: the organizational model defining the structure of the society, consisting of roles and interactions; the social model assigning roles to agents; and the interaction model describing possible interactions between agents. Thus, the OperA framework addresses both organizational structure and dynamics. The internal representation of agents is not clearly defined in this framework.

FORMAL FOUNDATIONS OF THE PROPOSED FRAMEWORK

The proposed framework introduces four interrelated views: performance-oriented, process-oriented, organization-oriented, and agent-oriented. The first-order sorted predicate logic serves as a formal basis for defining dedicated modeling languages for each view. These languages provide high expressivity for conceptualizing a variety of concepts and relations using sorts, sorted constants, variables, functions and predicates. Furthermore, these languages allow expressing both quantitative and qualitative aspects of different views.

To express temporal relations in specifications of the views, the dedicated languages of the views are embedded into the Temporal Trace Language (TTL) (Sharpanskykh, 2008), a variant of the order-sorted predicate logic. In TTL the organizational dynamics are represented by a trace, i.e. a temporally ordered sequence of states. Each state is characterized by a unique time point and a set of state properties that hold. State properties are specified using the dedicated language(s) of the view(s). In TTL, formulae of the state language are used as objects. For enabling dynamic reasoning TTL includes special sorts: TIME (a set of linearly ordered time points), STATE (a set of all state names of an organization), TRACE (a set of all trace names), and STATPROP (a set of all state property names).

A state for an organization is described by a function symbol state of type $\mathsf{TRACE}\times\mathsf{TIME}\rightarrow\mathsf{STATE}$.

The set of function symbols of TTL includes:

$\wedge,\vee,\rightarrow,\leftrightarrow: \mathsf{STATPROP}\times\mathsf{STATPROP}\rightarrow\mathsf{STATPROP}$,

$\mathsf{not}: \mathsf{STATPROP}\rightarrow\mathsf{STATPROP}$,

$\forall,\exists: \mathsf{SVARS}\times\mathsf{STATPROP}\rightarrow\mathsf{STATPROP}$,

which counterparts in the state language are Boolean propositional connectives and quantifiers. States are related to names of state properties via the formally defined satisfaction relation denoted by the infix predicate |=, state(γ,*t*)|=*p*, which denotes that the state property with a name *p* holds in trace γ at time point *t*. Both state(γ,*t*) and *p* are terms of TTL. In general, TTL terms are constructed by induction in a standard way from variables, constants and function symbols typed with all before mentioned TTL sorts. Transition relations between states are described by dynamic properties expressed by TTL-formulae. The set of atomic TTL-formulae is defined as:

(1) If v_1 is a term of sort STATE, and u_1 is a term of the sort STATPROP, then $v_1|=u_1$ is an atomic TTL formula.

(2) If τ_1, τ_2 are terms of any TTL sort, then $\tau_1=\tau_2$ is an atomic TTL formula.

(3) If t_1, t_2 are terms of sort TIME, then $t_1<t_2$ is an atomic TTL formula.

The set of well-formed TTL-formulae is defined inductively in a standard way using Boolean propositional connectives and quantifiers. TTL has semantics of the order-sorted predicate logic. A more detailed specification of the syntax and semantics of the TTL is given in (Sharpanskykh, 2008).

A set of structural and behavioral *constraints* imposed on organizational specifications can be identified. Formally, this set is represented by a *logical theory* consisting of formulae constructed in the standard predicate logic way from the terms of the dedicated language of the views (and of TTL if temporal relations are required). A specification is *correct* if the corresponding theory is satisfied by this specification, i.e., all sentences in theory are true in the logical structure(s) corresponding to the specification. The constraints are divided in two main groups: generic and domain-specific. *Generic constraints* define general restrictions for a view or the whole specification that need to be satisfied by the specification of every organization. Two types of generic constraints are considered. *Structural integrity and consistency constraints* are based on the rules of specification composition, e.g. constraints guaranteeing the internal consistency of a structure within a particular view. The *constraints imposed by the physical world* are not dictated by the framework but by the rules of the physical world which render certain situations impossible, e.g. an agent cannot be at two different physical locations at the same time. The set of generic constraints is predefined and can be reused for every specification.

Domain-specific constraints are dictated by the application domain and may be added or modified by the designer. They express facts and restrictions valid in the particular application domain but not necessarily in other domains. Domain-specific constraints can be *imposed by the organization* itself, *external parties*, e.g. the government, the society, other companies, as well as by *the physical world* of the specific application domain. Domain-specific constraints may or may not be directly reusable for other companies in the same or other domains. To support the process of designing such constraints, templates can be provided with parameters that can be customized by the designer.

INTRODUCTION TO THE CASE STUDY

The proposed framework was applied for modeling and analysis of an organization from the security domain within the project CIM (Cybernetic Incident Management, http://www.almende.com/cim/). This organization has many features of a mechanistic organization: in particular, hierarchically linked job positions with clear responsibilities that use standard well-understood technology. The formal documents of the organization provide diverse details about the structure and dynamics of the organization, which need to be formally represented in a feasible model of the organization. In particular, the following aspects should be considered: performance indicators (PIs) and goals; processes, resources and flows of processes; roles (positions) and different types of relations between them (e.g., interaction, power);

characteristics and behavior of organizational agents allocated to roles (positions). Furthermore, relations between different aspects should be modeled as well (e.g., between tasks and goals; between roles and agents). The following Sections illustrate how the proposed framework can be used to create formal specifications for particular perspectives of the organization. These specifications can be further used to perform automated analysis.

The main purpose of the organization in focus is to deliver security services (surveillance, consultancy, training, etc.). The organization has a well-defined multi-level structure that comprises two divisions over several areas. The company employs approximately 230.000 persons with predefined job descriptions. The examples given in this Chapter are related to the planning of the assignment of security officers to customer locations. The planning process consists of forward (long-term) planning and short-term planning. Forward planning is the process of creation, analysis and optimization of forward plans for the allocation of security officers within the organization based on customer contracts. It is performed by a team of forward planners from the forward planning group under the supervision of the Manager Planning. The forward planning group is centralized for the organization as part of the department Operations Support and reports to the Operations Support Manager. During short-term planning, plans describing the allocation of security officers within certain area for a week are created and updated based on the forward plan and up-to-date information. Also as part of short-term planning, daily plans are created based on short-term plans and information coming from the security employees about their availability and other changes via data change forms. For each area, the short-term planning is performed by the area planning team led by its Team Leader. During short-term planning the planners can get supervision, advice or information from forward planners, depending on the specific circumstances. The planners also communicate with the area's Unit Manager who is in charge of the security employees within this area, takes care of collecting and processing the data change forms and supervises the daily plans execution.

In both types of planning, resources are used including the company's personnel database, customer contracts, data change forms, plans and other paper-based or electronic resources. Other activities of the planners include reporting to the management team, training for improving qualifications, evaluations, etc.

MODELING VIEWS

In this section, the views of the proposed framework are presented. Three of them, process-oriented, performance-oriented and organization-oriented, have prescriptive character and define the desired behavior of the organization. The fourth view, agent-oriented, describes and integrates agents into the framework.

The process-oriented view describes static hierarchies of organizational tasks and resources as well as flows of processes (workflows), and relations between these structures. *The performance-oriented view* defines the goals hierarchies of the organization, the relevant PIs and their relationships. Within *the organization-oriented view*, organizational roles, their interaction, authority, responsibility and power relations are defined. In *the agent-oriented view*, different types of agents with their capabilities are identified and principles for allocating agents to roles are formulated. The four views are connected via relations. Each view and the relations between views are discussed in the following subsections.

Process-oriented view

The process-oriented view contains information about the organizational functions, how they are related, ordered and synchronized and the resources they use and produce. The main concepts are: task, process, workflow, resource type and resource which, together with the relations

between them, are specified in the formal language L_{PR} (Sharpanskykh, 2008). A *task* represents a function performed in the organization and is characterized by *name*, *maximal_duration* and *minimal_duration*. Table 1 gives examples of tasks considered in the case study related to planning of the assignment of security officers to locations. The tasks were extracted from available company documents on procedures and job descriptions (as discussed later in this Chapter). Column 1 contains the identification numbers assigned to the tasks for convenient reference. Column 2 contains the names of the tasks reflecting their content. Columns 3 and 4 list the resources used and produced by the tasks.

Tasks can range from very general to very specific. General tasks can be decomposed into more specific ones using AND- and OR-relations forming hierarchies. Fig. 1 shows the refinement relations of task planning from the case study. The numbers correspond to column 1 of Table 1 and reflect the position of the task in the hierarchy. For example task process_new_data_forms (4.1) describes the processing (correcting and summarizing) of forms from the security officers containing information about changes in their availability. This is needed in the planning process to produce an up-to-date plan taking into account the availability of the security officers, therefore 4.1 is a subtask of the overall planning task 4.

Table 1. Tasks from the case study

Number	Name	Uses resource types	Produces res. types
4	planning		
4.1	process_new_data_change_forms	data change forms	analysis results of data change forms
4.2	create_and_inform_correct_and_optimized_shortterm_plan		
4.2.1	create_shortterm_plan		
4.2.1.1	shortterm_plan_creation_discussion	forward plan	information about a decision on assignment of a task of creating a short-term plan
4.2.1.2	estimate_human_capacity_per_location	forward plan, personnel data, customer order details, analysis results of daily change forms	data about available and required human capacity per location
4.2.1.3	assign_officers_to_tasks	forward plan, personnel data, the short-term planning procedure, data about available and required human capacity per location	information about the assignment of security officers
4.2.1.4	input_planning_data	planning software handbooks, the short-term planning procedure, information about the assignment of security officers	short-term plan for next month
4.2.2	check_and_improve_shortterm_plan	short-term plan for next month, the short-term planning procedure	correct short-term plan for next month
4.2.3	optimize_shortterm_plan	correct short-term plan for next month	correct optimized short-term plan for next month
4.2.4	inform_all_concerned_about_shortterm_plan	correct optimized short-term plan for next month	correct optimized short-term plan for next month
4.3	create_and_inform_daily_plan		

Each task can be instantiated in one or more processes in a workflow. A *workflow* is defined by a set of (partially) temporally ordered *processes*. Each process, except for the special ones with zero duration introduced below, is defined using a task as a template and all characteristics of the task are inherited by the process. Decisions are also treated as processes that are associated with decision variables taking as possible values the possible decision outcomes.

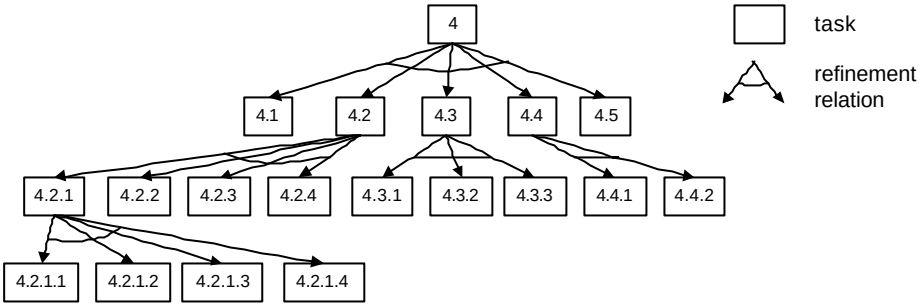


Fig. 1 A part of the task hierarchy from the case study

A workflow starts with the process BEGIN and ends with the process END; both have zero duration. The (partial) order of execution of processes in the workflow is defined by sequencing, branching, loop and synchronization relations. A *sequencing relation* starts_after expresses temporal-order precedence of processes. *Synchronization relations* define temporal relations between processes that are executed in parallel (starts_with, finishes_with, starts_during). *Branching relations* define and- and or-structures. An and(or)-structure with name *id*, starts with the zero-duration process begin_and(*id*) (begin_or(*id*)) and finishes by the zero-duration process end_and(*id*) (end_or(*id*)) represented graphically by rhombuses. Our treatment of and-structures is similar to the parallel split pattern combined with all types of the merge pattern from (Van der Aalst *et al.* 2003), represented here by an and-condition determining when the process following the and-structure may start. Our treatment of or-structures allows realizing both exclusive and multiple-choice patterns from (Van der Aalst *et al.* 2003). For every or-structure a condition is defined to determine which branches of the or-structure will start. *Loop relations* are defined over *loop*-structures with conditions that realize the cycle patterns from (Van der Aalst *et al.* 2003). A loop-structure *id*, starts with begin_loop(*id*) and finishes by end_loop(*id*). A Boolean condition and the maximal number of executions of every loop are specified.

The processes and the relations between them, expressed in L_{PR}, can be (partially) visualized as in Figures 2 and 3 at different levels of abstraction from very high-level (aggregated) to very detailed and specific level. Detailed levels of abstraction are achieved by refining processes into more specific ones using the refinement relations in the corresponding task hierarchies.

Within the case study, Fig. 2 presents the workflow that takes an aggregated view on the processes of daily planners performed during a contract period. Such processes as planning and check_plan_conformity can further be refined into more specific workflows using the identified hierarchy of tasks. Fig. 3(a) takes an aggregated view on the process planning, whereas Fig. 3(b) provides more specific details of this process by refining some of processes (i.e., provide_correct_data_change_forms_to_planners, update_and_inform_shortterm_plan and create_and_inform_daily_plan). The beginning of the workflow in Fig. 3(b) can be expressed in L_{PR} as follows:

starts_after(process_new_data_change_forms,BEGIN,0)
starts_after(begin_and(and1),process_new_data_change_forms,0)
starts_after(generate_daily_planning_data,begin_and(and1),0)
starts_after(begin_or(or1),begin_and(and1),0),…

Tasks use, consume or produce resources of different types. Resource types represent tools, supplies, components, data, etc. and are characterized by *name*, *category* (discrete, continuous), *measurement_unit*, *expiration_duration* (the length of the time interval when a resource type can be used). Resource types can sometimes be decomposed forming resource hierarchies, e.g. a database can consist of several data tables containing different information that can be used separately by different tasks, while other tasks might require the whole database. A resource type can have different function (purpose) from the resource types in its decomposition, e.g. a car has a different purpose from each of its components.

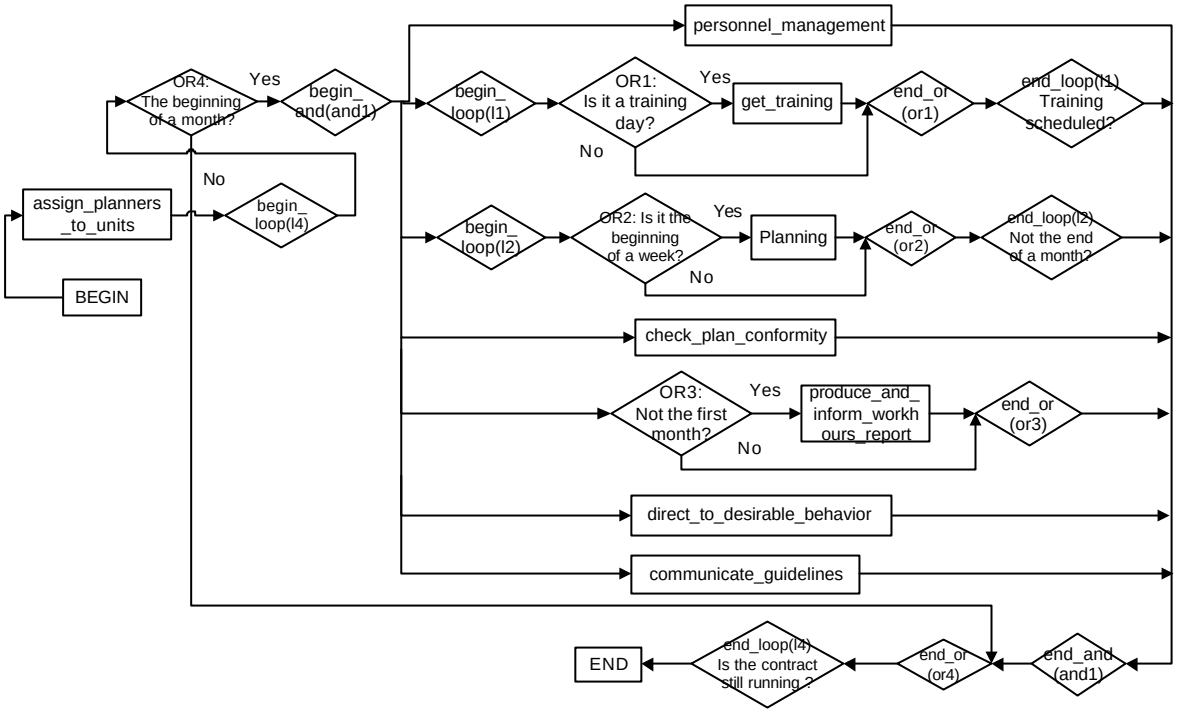


Fig. 2 The workflow that takes an aggregated view on the activities of daily planners performed during a contract period

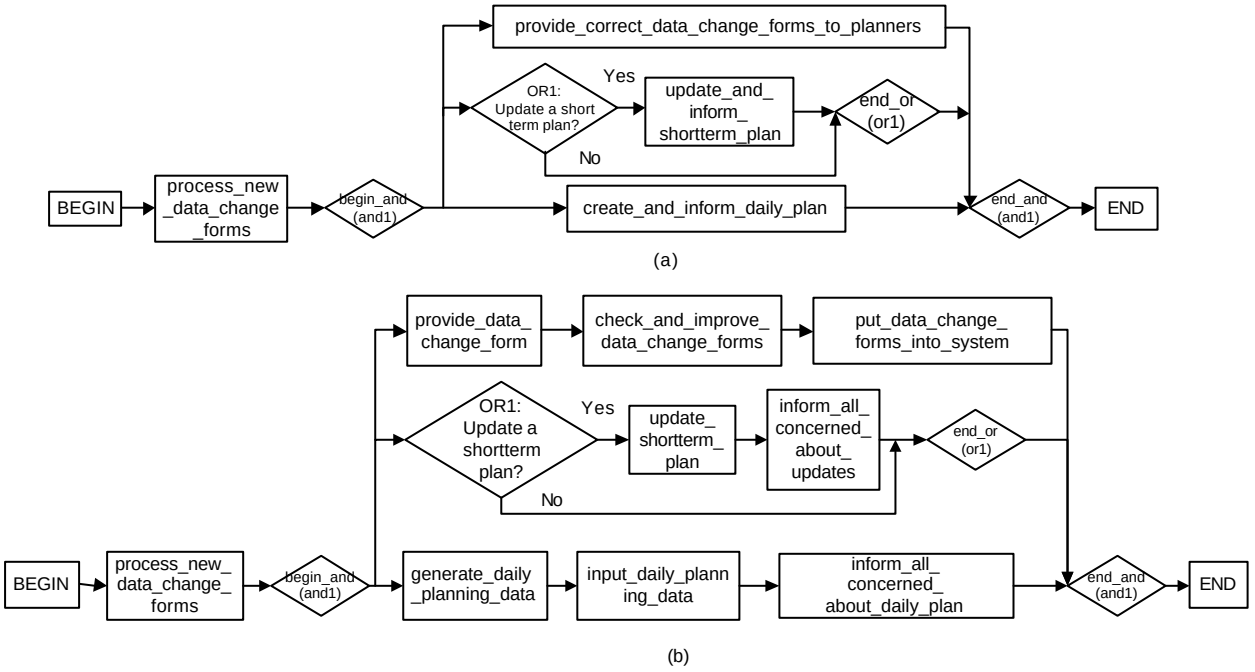


Fig. 3 An aggregated view (a) and a detailed view (b) on the planning process

Resources are instances of resource types and inherit their characteristics, having, in addition, *name* and *amount*. Resources are used, consumed or produced by processes in the workflow. Multiple resources of the same resource type can be produced by different processes and are differentiated by their names.

Some resources can be shared, used simultaneously, by a set of processes (e.g., storage facilities, transportation vehicles). Sets of processes can be defined that are allowed to share specific resources. Our representation of shared resources differs from (Barkaoui & Petrucci,

1998) in several aspects: (1) the shared resource amount is used by processes simultaneously; (2) alternative sets of processes that are allowed to share a resource can be defined; (3) different amounts of a resource can be shared simultaneously; (4) specific conditions (requirements) for resource sharing can be defined. In the case study a short-term plan can be shared between the tasks inform_all_concerned_about_shortterm_plan and create_and_inform_daily_plan.

In the case study a number of resource types were identified, some listed in Table 1 in relation to the lowest-level tasks. Most of the identified resources are discrete (a forward plan, data change forms, customer order details). The resources used/consumed/produced by a higher-level task comprise the resources used/consumed/produced by the lower-level tasks in its refinement.

Sometimes it is important to monitor where the resources are at certain time points, for which the concept *location* is used e.g. representing the available storage facilities. Processes can add or remove resources of certain types from locations, which can be specified using predicates from the L_{PR} language (process_rem_resource_type_from, process_adds_resource_type_to: p:PROCESS × r:RESOURCE_TYPE × l:LOCATION).

The generic constraints for this view include structural constraints on the correctness of workflows, task and resource hierarchies and constraints from the physical world. For hierarchies, consistency should be maintained by making sure the set of inter-level constraints is satisfied. Consider two examples of such interlevel structural generic constraints:

GC_PO1: For every and-decomposition of a task, the minimal duration of the task is at least the maximal of all minimal durations of its subtasks.
Formally, $\forall t$:TASK, t_1:TASK, L:TASK_LIST $\forall tp,\gamma$
state(γ,tp)|=[is_decomposed_to(t,L) $\wedge$ is_in_task_list(t_1,L) $\rightarrow$ t.min_duration=t_1.min_duration]

GC_PO2: If a task uses certain resource type as input then there exists at least one subtask in at least one and-decomposition of this task that uses this resource type.
Formally, $\forall t$:TASK, rt:RESOURCE_TYPE, L:TASK_LIST, v:VALUE, $\forall tp,\gamma$
state(γ,tp)|=[task_uses(t,rt,v) $\wedge$ is_decomposed_to(t,L) $\rightarrow$ $\exists t_1$:TASK, L_1:TASK_LIST, v_1:VALUE is_decomposed_to(t,L_1) $\wedge$ is_in_task_list(t_1,L_1) $\wedge$ task_uses(t_1,rt,v_1)]

Further, consider examples of physical world generic constraints:

GC_PO3: For every process that uses certain amount of a resource of some type as input, without consuming it, either at least that amount of resource of this type is available or can be shared with another process at every time point during the possible execution of the process.

GC_PO4: Non-sharable resources cannot be used by more than one process at the same time

Domain-specific constraints may or may not be directly reusable for the specification of other organizations in the same or other domains. Therefore to support the process of designing such constraints pre-specified templates expressed in L_{PR} can be used. Such templates, created by analysts skilled in logic for a particular organization, may be reused multiple times by different instantiations. A template is instantiated by assigning specific values to its parameters. Such reuse can reduce the effort for specifying domain-specific constraints for every organization. It also requires less expertise in logic than for the specification of a new constraint in formal language. An example of a template with its parameter in brackets is:

DC_PR1(*rt*:RESOURCE_TYPE): Resource of type rt is produced in the workflow.
Formally, $\exists p$:PROCESS, t:TASK, r:RESOURCE, v:VALUE $\forall tp,\gamma$ state(γ,tp)|=[process_output(p,r) $\wedge$ is_instance_of(p,t) $\wedge$ is_instance_of(r,rt) $\wedge$ task_produces(t,rt,v)]

For the case study this constraint may be instantiated e.g. into DC_PR1(daily_plan) meaning that a daily plan should be produced in the workflow.

Formally:

$\exists p$:PROCESS, t:TASK, r:RESOURCE, v:VALUE, $\forall tp,\gamma$ state(γ,tp)|=[process_output(p,r) $\wedge$ is_instance_of(p,t) $\wedge$ is_instance_of(r,daily_plan) $\wedge$ task_produces(t,daily_plan,v)]

It can also be instantiated for a short-term plan and for a forward plan as well as for other resources not directly related to planning.

Performance-oriented view

Central notions in the performance-oriented view are goal and PI. Many organizations nowadays set goals to be achieved reflecting different aspects of the organizational performance. The goals are evaluated by monitoring and analyzing related PIs. Traditionally only numerical (often financial) PIs were considered such as costs, profits, number of clients, however today it is considered important to also monitor indicators such as customer satisfaction, employees' motivation which are qualitative and difficult to assess. Modeling of goals is supported to a various degree by a number of existing frameworks in enterprise modeling. However the concept of a PI has been underrepresented in the literature. Our approach differs in explicitly representing PIs, the link between a goal and the PI that measures its satisfaction and the relationships between PIs that can be used for reasoning at the design phase (discussed in the Section on methodological aspects). Here the concepts of the performance-oriented view are defined.

A PI is a quantitative or qualitative indicator that reflects the state or progress of the company, unit or individual. The characteristics of a PI include:

name;

definition;

type (continuous, discrete);

measurement *unit*;

time_frame (the length of the time interval for which it will be evaluated, e.g. per day, per month);

measurement *scale* (e.g. low-med-high);

min value;

max value;

source – the internal or external source used to extract the PI (company policies, mission statements, business plan, job descriptions, laws, domain knowledge, etc.);

owner (the performance of which role or agent it measures/describes);

threshold – the cut-off value separating changes in the value of the PI considered small and changes considered big, used to define the degree of causal influence between PIs (see below the discussion on relationships between PIs);

hardness – soft or hard, where soft means not directly measurable, qualitative (customer's satisfaction, company's reputation, employees' motivation), and hard means measurable, quantitative (number of customers, time to produce a plan).

For the part of the case study related to planning, using company documents, 33 PIs were identified, among which:

Name: PI5
Definition: average correctness of plans
Type: discrete
Time_frame: month
Scale: very_low-low-med-high-very_high
Source: mission statement, job descriptions
Owner: forward/short-term planning departments
Threshold: 2 units
Hardness: soft

Name: PI27
Definition: time to create new short-term plan
Type: continuous
Time_frame: month
Scale: REAL
Min_value: 0
Max_value: max_time_CST
Unit: hour
Source: job descriptions
Owner: short-term planning departments
Threshold: 24h
Hardness: hard

PIs can be related through various relationships. The following are considered in the framework: (strong) positive/negative causal influence of one PI on another, positive/negative correlation between PIs, aggregation (two PIs express the same measure at different aggregation levels). In deciding whether a causal influence is strong or not, the threshold characteristic of the PIs is used. If change above the threshold is observed then the relationship is considered strong. Relationships can be identified using e.g. company documents, domain knowledge, inference from known relations, statistical or data mining techniques, knowledge from other structures of the framework. Using these relations, a graph structure of PIs can be built. The PI structure for the case study discussed here is given in (Sharpanskykh, 2008) as well as more details on PIs and their relationships.

Based on PIs, PI expressions can be defined as mathematical statements over PIs that can be evaluated to a numerical, qualitative or Boolean value, e.g. PI5=high, PI5=medium, PI27<8h but also PI5, PI27. PI expressions are used to define goal patterns. The *type* of a goal pattern indicates the way its property is checked: *achieved (ceased)* – true (false) for a specific time point; *maintained* (*avoided*) – true (false) for a given time interval; *optimized* – if the value of the PI expression has increased, decreased or approached a target value for a given interval. Some of the possible goal patterns that can be defined on the PIs and PI expressions given above are: "maintained PI5=high", "avoided PI27<8h", "increased PI5", etc.

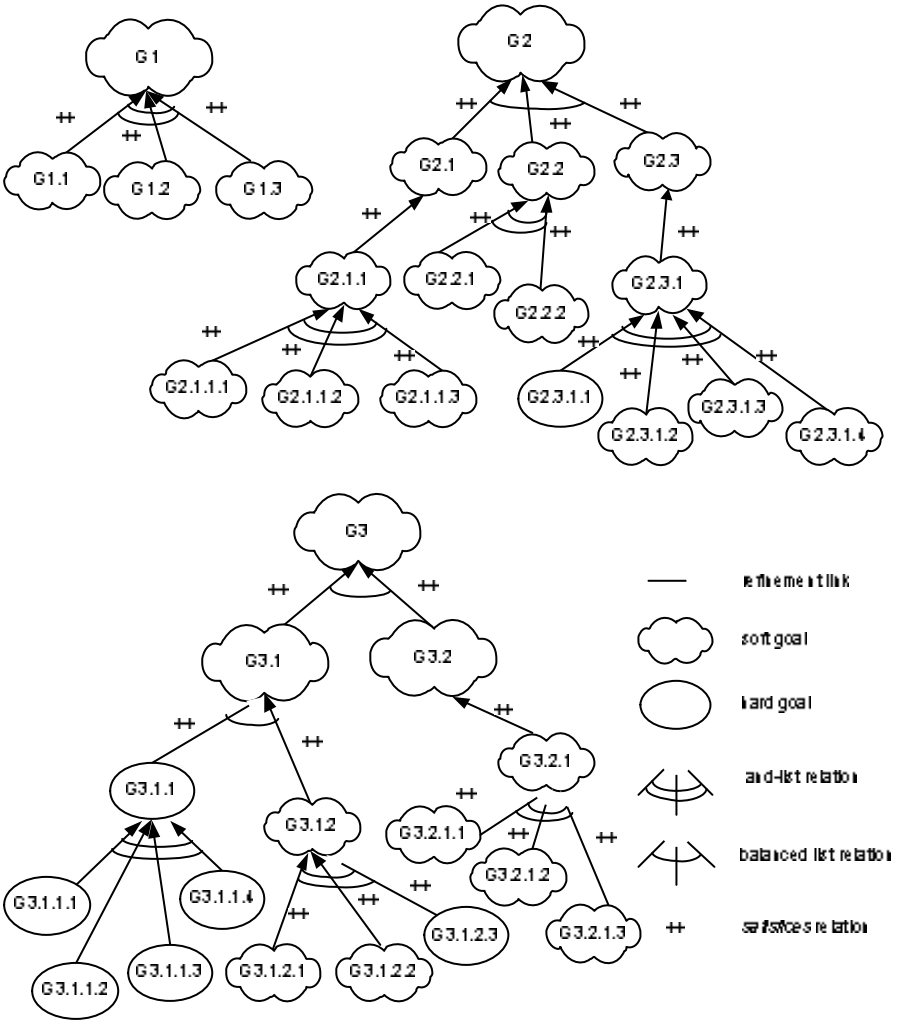
Goals are objectives that describe a desired state or development and are defined by adding to goal patterns information on desirability and priority. The characteristics of goals include:

name;
definition;
priority;
evaluation_type – achievement goal (based on 'achieved' or 'ceased' goal pattern; should be evaluated for a time point) or development goal (based on 'maintained', 'avoided' or 'optimized' goal pattern; should be evaluated for a time interval);
horizon – for which time point/interval should the goal be satisfied;
ownership – organizational or individual;
perspective (for organizational goals) – which point of view the goal describes, of management, supplier, customer, or society;
hardness – hard (satisfaction can be established, the goal can be either satisfied or failed) or soft (satisfaction cannot be clearly established, instead degrees of *satisficing* are defined – the goal can be weakly or strongly satisficed or denied);
negotiability – is the goal negotiable when conflicts with other goals should be solved/avoided.

Examples of goals identified for the case study are:

Name: G3.2
Definition: It is required to maintain high efficiency of allocation of security officers
Priority: high
Horizon: long-term
Evaluation_type: development
Ownership: organizational
Perspective: management, customer
Hardness: soft
Negotiability: negotiable

Name: G3.1.1.1
Definition: It is required to achieve that the time to update a short-term plan given operational data is =48h
Priority: high
Horizon: short-term
Evaluation_type: achievement
Ownership: organizational
Perspective: management
Hardness: hard
Negotiability: negotiable



Goal can be refined into sub-goals forming hierarchies. Information about the satisfaction of lower-level goals can be propagated to determine the satisfaction of higher-level goals. A goal can be refined into one or more alternative lists of goals of AND-type or balanced-type (a more fine-tuned way of decomposition inspired by the weighted average function) (Sharpanskykh, 2008). For each type, specific propagation rules are defined to determine the satisfaction value (for hard goals) or the level of satisficing (for soft goals) of the higher-level goal. Fig. 4 shows the goals hierarchy for the forward and the short-term planning in our case study.

Using the concepts and relations of the performance-oriented view, constraints can be formulated. An example of a generic structural constraint is: "If two PIs are related by aggregation relation, they should have the same type and measurement unit". Formally:

$\forall pi_1,pi_2$:PI, $\forall tp,\gamma$ state(γ,tp)|=[aggregation_of(pi_1,$pi2$) $\rightarrow$ pi_1.unit=pi_2.unit $\wedge$ pi_1.type=pi_2.type]

Organization-oriented view

In the organization-oriented view organizations are modeled as composite roles that can be refined into composite or simple roles, representing as many aggregation levels as needed. The refined role structures correspond to different types of organization constructs (e.g., groups, units, departments). The view provides means to structure and organize roles by defining interaction and power relations on them. First, interaction relations are discussed.

Each role has an input and an output interface, for the interaction with other roles and the environment. Role interfaces are described in terms of interaction (input and output) ontologies: signatures specified in order-sorted logic. Generally speaking, an input ontology determines what information is allowed to be transferred to the input of a role (or the environment), and an output ontology predefines what information can be generated at the output of a role (or the environment). To specify a type of interaction (e.g., communication), the ontologies of both interacting roles (role-source of interaction r_1 and role-destination of interaction r_2) should include the predicate:

communicate_from_to(r_1:ROLE,r_2:ROLE,s_act:SPEECH_ACT,$message$:STRING),

where s_act is a speech act (e.g., inform, request, ask) and $message$ is the content. Roles of the same aggregation level, allowed to interact, are connected by an interaction link indicating the direction of interaction. To represent information transition between roles of two adjacent aggregation levels (e.g., between a role representing a department and a role that belongs to this department), interlevel links are used. In Fig. 5-7 communication relations between the roles of the organization in focus are provided. These relations correspond to the communication channels that exist (e.g., specified explicitly) in the organization and may be used in different organizational scenarios. In particular, Fig. 5 shows the interaction relations between the roles in the organization at the first aggregation level. Fig. 6 presents the relations within Area role (a) and within Operations_Support role (b) at the second aggregation level and the relations between the subroles of the roles Forward_Planning (c) and Team_Planning (d) at the third aggregation level. By using different levels of abstraction, scalability of graphical representation is achieved.

The representation of the environment may vary in organizational specifications. It can be defined by a set of objects with certain properties and states and by causal relations between objects. In other cases the dynamics of the environment is described by (high-level) processes and trends (e.g. changes of the market situation, natural environmental oscillations).

Interaction relations at the generalized level, represent templates that can be instantiated for a particular case. For example, the documents of the organization from the case study define standard patterns of interaction between the Forward_Planner and Manager_Planning roles that can be modeled at the generalized (template) level. However, for a more detailed analysis of the organizational dynamics, a more specific representation defining interaction relations between particular role instances of Forward_Planner role (e.g., from different planning teams) is needed (see Fig. 7).

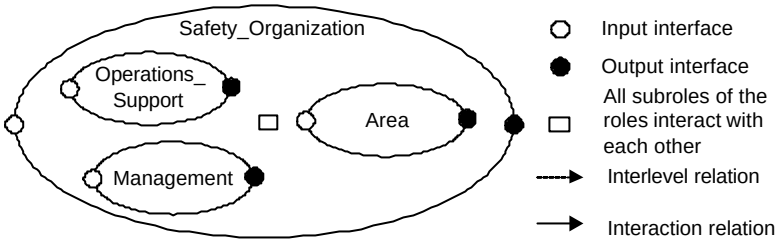


Fig. 5 Interaction relations between the roles of the Safety_Organization considered at the first aggregation level

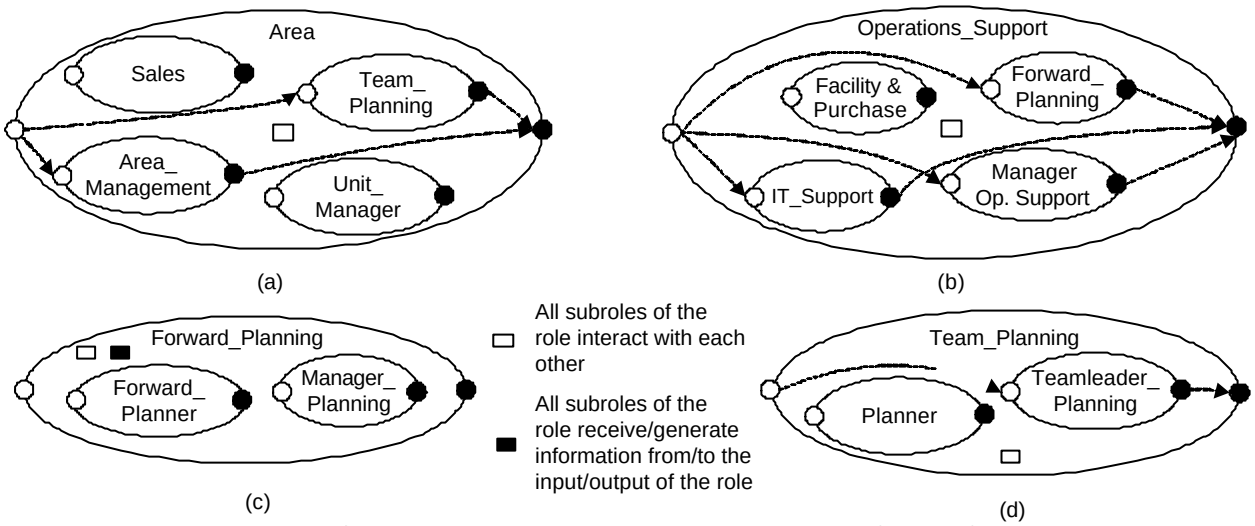


Fig. 6 Interaction relations of Area (a) and Operation_Supports (b) roles of the Safety_Organization considered at the second aggregation level and the Forward_Planning (c) and Team_Planning (d) roles of the Safety Organization considered at the third aggregation level

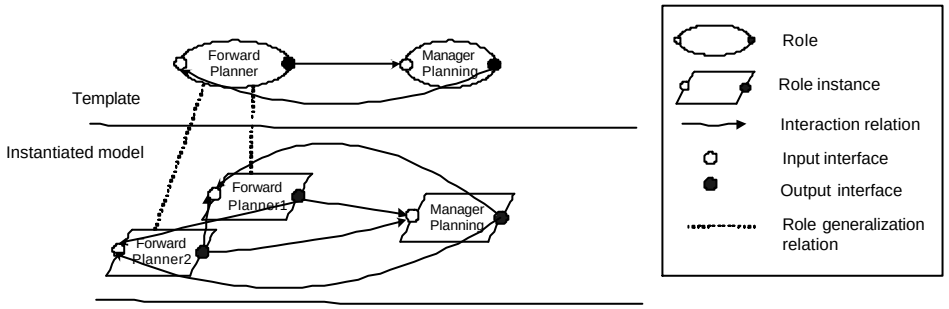


Fig. 7 The graphical representation of interaction relations between the roles Forward_Planner and Daily_Planner (the template) and their instances (the instantiated model)

The treatment of agent interaction in our framework differs from the agent interaction design approach described in Chapter 8 "Hermes: A Pragmatic Approach to Flexible and Robust Agent Interaction Design and Implementation" by C. Cheong and M. Winikoff. In this approach interaction goals are defined for agents that can be achieved by performing interaction acts. In our framework goals are related to interaction relations through processes that require this interaction.

Power relations on roles also constitute a part of the formal organizational structure. Roles may have different rights and responsibilities w.r.t. different aspects of task execution, such as execution, passive monitoring, consulting, making technological decisions (i.e., decisions on technical questions related to task content) and making managerial decisions (i.e., decisions on general issues related to the task). To specify responsibilities of roles the following relation is used: is_responsible_for: r:ROLE × $aspect$:ASPECT × a:TASK, task a is under responsibility of role

r w.r.t. *aspect*. The responsibility relations for some tasks from Table 1 are as follows. For task process_new_data_change_forms the role Daily_Planner is responsible for task execution and the role Area_Manager is responsible for making technological and managerial decisions. For task create_and_inform_correct_and_optimized_shortterm_plan the role Forward_Planner is responsible for all aspects and role Teamleader_Planning is jointly responsible for execution. For task create_and_inform_daily_plan Teamleader_Planning is responsible for all aspects and Daily_Planner is jointly responsible for task execution.

The role granting responsibility for certain aspects of a task to another role should be (1) responsible for making managerial decisions w.r.t. this task; (2) superior of the role w.r.t. the task. Superior-subordinate relations w.r.t. organizational tasks are specified by: is_subordinate_of_for: r_1:ROLE × r_2:ROLE × a:TASK. In the case study a number of superior-subordinate relations were identified, including:

is_subordinate_of_for(Forward_Planner,Manager_Planning,create_and_inform_correct_longterm_plan),
is_subordinate_of_for(Daily_Planner,Teamleader_Planning,create_and_inform_daily_plan).

Control over resources for roles is specified using the relation has_control_over: r_1:ROLE × *res*:RESOURCE.

Some of the generic constraints that define the consistency and integrity of any role interaction structure have been identified in this view:

GC_OO1: No role can be a subrole of itself at any aggregation level.
GC_OO2: Each subrole of a composite role *r* should interact with at least one other subrole of *r*.
Formally,
In the organizational structure Γ $\forall r_1$:ROLE, $\forall tp,\gamma$ state(γ,*tp*)|=[subrole_of_in(r_1,*r*,Γ) → $\exists r_2$:ROLE $\exists e$:INTERACTION_LINK subrole_of_in(r_2,*r*,Γ) ∧ (connects_to(*e*,r_2,r_1,Γ) ∨ connects_to(*e*,r_1,r_2,Γ))]

Another set of generic consistency constraints is formulated over authority relations of the organization based on the literature from Social Science, among which:
GC_OO3: Only roles that have the responsibility to make managerial decision w.r.t. some process are allowed to authorize other roles for some aspect of this process

Formally,
$\forall r_1,r_2$:ROLE, $\forall a$:PROCESS, $\forall asp$:ASPECT, $\forall \gamma,tp$
state(γ,*tp*)|=[authorizes_for(r_1,r_2,*asp*,*a*) → is_responsible_for(r_1,manage_des,*a*)]

Some of the generic constraints ensure the consistency between the interaction and authority structures, e.g.:

GC_OO4: Roles related by a superior-subordinate relation should interact.
Formally,
$\forall r_1,r_2$:ROLE $\forall a_1$:PROCESS, $\forall tp,\gamma$ state(γ,*tp*)|=[is_subordinate_of_for(r_2,r_1,a_1) →
$\exists e_1,e_2$:INTERACTION_LINK connects_to(e_1,r_2,r_1,Γ) ∧ connects_to(e_2,r_1,r_2,Γ)]
GC_OO5: The role supervising the execution of a process should interact with the role performing the process.

Further, consider an example of a domain-specific constraint, particular instances of which are defined for the case study:

DC_OO1(*r*1:ROLE,*r*2:ROLE,information_type): Particular information should be transferred between specific roles.

In the case study a correct and optimized short-term plan should be provided to all concerned parties, among which is Unit_Manager. To represent such communication, an instantiated version of this constraint is used:

DC_OO1(Forward_Planner,Unit_Manager,correct_and_optimized_short_term_plan).

Agent-oriented view

To create plausible organization models, in addition to formal (prescriptive, documented) aspects, also informal aspects of human behavior should be considered. Models of agents defined in the agent-oriented view are based on psychological and social theories (e.g., work motivation theories described in (Pinder, 1998)).

For each role a set of requirements on agent capabilities (i.e., knowledge and skills) and personal traits is defined. Requirements on knowledge define facts and procedures, confident understanding of which is required from an agent. Skills describe developed abilities of agents to use effectively and readily their knowledge for tasks performance. In the literature four relevant types of skills are distinguished: technical, interpersonal, problem-solving/decision-making and managerial skills. To enable testing (or estimation) of skills and knowledge, every particular skill and knowledge is associated with a PI (e.g., the skill 'typing' is associated with the PI "the number of characters per minute"). Personal traits are divided into five categories formulated in psychology (De Raad & Perugini, 2002): openness to experience, conscientiousness, extroversion, agreeableness, and neuroticism. Agent capabilities and traits can have different levels of importance. Whereas the required for a role capabilities and traits are compulsory for taking the role, desired capabilities and traits considered as an advantage.

In the case study, the role Daily_Planner requires the agent to have knowledge and technical skills related to daily planning, as well as some interpersonal skills. The company also defined requirements on personal traits related to conscientiousness (self-discipline, responsibility, aim for achievement) and agreeableness (cooperative work style).

In general, the efficiency of allocation of an agent to a role is dependant on how well the agent's characteristics fit with the role requirements. However, modern organizations implement very diverse allocation principles (e.g., based on equality, seniority or stimulation of novices). Such principles can be formalized as allocation policies comprising TTL properties. In the case study the standard allocation policy was used: an agent is allocated to a role if s/he possesses the necessary capabilities and traits defined as the requirements for the role.

To model the dynamics of an agent in organizational context, the agent's intentional and motivational aspects are considered. In modern social science behavior of individuals is considered as goal-driven. It is recognized that high-level goals of individuals are dependant on their needs. Currently the following division of needs is identified in social science: *extrinsic needs* associated with biological comfort and material rewards; *social interaction needs* that refer to the desire for social approval, affiliation and companionship; *intrinsic needs* that concern the desires for self-development, self-actualization, and challenge.

In modern organizations when an individual is allocated to a role, the identification of his/her specific lower-level goals is performed in cooperation with a managerial representative of the organization. During this process, the high-level goals, based on the agent's needs are refined into more specific goals aligned with organizational goals using AND- and OR-relations. Often two types of such goals are distinguished: development (or learning) and performance goals. Development goals reflect wishes of agents to gain certain knowledge or some skills that are also useful for the organization. Performance goals usually concern the effectiveness and efficiency of the execution of the tasks already allocated to the agent. Both development and performance goals are formalized using the language of the performance-oriented view and may change over time.

The motivation of agents to perform certain tasks is important to ensure the satisfaction of both individual and organizational goals related (directly or indirectly) to these tasks. Therefore, the motivational aspect of the agent behavior should be explicitly represented in the models of agents. The highest motivation is demonstrated by an agent w.r.t. actions (e.g., the execution of organizational tasks) that (significantly) contribute to the satisfaction of his/her primary goals. For reasoning about agents' motivation and work behavior, Vroom's version of the expectancy

theory (Pinder, 1998) is used which establishes causal dependencies between a number of individual, organizational and environmental parameters and the agent's motivation to perform certain actions (processes). The expectancy theory is one of the few organization theories that can be made operational and used for simulation.

The framework does not adhere to particular agent architecture (e.g., BDI, KARO). In the general case the dynamics of an agent can be specified by dynamic properties expressed in TTL. Furthermore, using TTL existing agent modeling architectures can be represented.

Unlike the other three views which are prescriptive, the agent-oriented view has descriptive character. Therefore no constraints specific to this view can be defined. However it is possible to define constraints involving the agent-oriented view and one or more other views. For example, a constraint between the agent-oriented, the process-oriented and the organization-oriented views can be the following: "Every recruited agent who has not performed a role in the organization in the past should attend a corporate induction event within 3 months of starting".

Relations between the views

The views of the framework are connected to each other via relations between their concepts. For example roles are committed to organizational goals and agents can be committed to individual or organizational goals. Goals must be realizable by tasks defined in the process-oriented view, which are represented (instantiated) by processes in the workflow. PIs measure aspects of the execution of processes (duration, output, precision, correctness, etc.) and can be owned by (reflect the performance of) roles and agents. Since roles in the organization-oriented view can be specified at different levels of aggregation, a PI can be owned by a single employee, a group or team or even the whole organization.

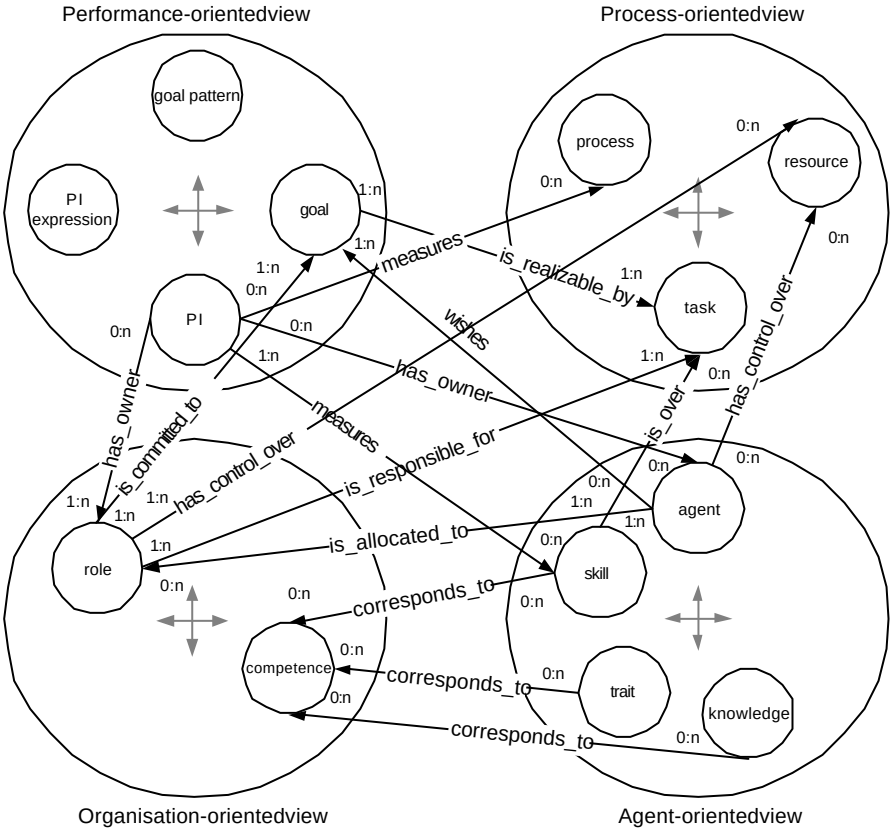


Fig. 8 Relations between the views of the framework

Roles are assigned responsibility for aspects of tasks execution such as technical or managerial decisions and execution. Agents described in the agent-oriented view can be assigned to roles for which different competences may be required. These competences can correspond to skills, traits and knowledge of agents. Skills are defined w.r.t. specific tasks and are measured by PIs. Also agents and roles may have control over resources.

The most essential relations and their cardinalities are depicted in Fig. 8 while the relations within the views are omitted.

Another way in which the views can be related is via constraints that can be expressed over concepts and relations from multiple views. Consider the following examples of generic constraints:

Over the process-oriented and organization-oriented views:
GC_MV1: At the beginning of each process for each of the basic aspects for this process (execution, tech_des, and manage_des) a responsible role should be assigned.

Over the performance-, process-oriented and organization-oriented views:
GC_MV2: If a role is committed to a goal, then this role should be responsible for some aspect(s) of a task(s) realizing this goal.
Formally,

$$\forall r{:}\text{ROLE}\ \forall g{:}\text{GOAL}\ \forall tl{:}\text{TASK\_LIST}\ \forall \gamma, tp$$
$$\text{state}(\gamma, tp) \models [\text{is\_committed\_to}(r,g) \wedge \text{is\_realizable\_by}(g,tl)$$
$$\rightarrow \exists a{:}\text{TASK}\ \exists asp{:}\text{ASPECT}\ \text{is\_in\_task\_list}(tl,a) \wedge \text{is\_responsible\_for}(r,asp,a)]$$

Consider a domain-specific constraint expressed over languages of multiple views:
DC_MV1(*r*:ROLE,*dr*:VALUE): The amount of working hours of each agent allocated to role *r* should not exceed *dr*

In this case study this constraint is instantiated into DC_MV2(Planner,8).

The relationships between the views also play a role in the process of organizational design with the framework. This is discussed in the next Section which is devoted to the methodology of using the framework.

METHODOLOGICAL ASPECTS

This Section describes how the framework can be used in practice, how the design process can be approached, structured and ordered, what issues need to be addressed, etc., depending on what information is available. First the general guidelines are discussed and then it is shown how they were used for the case study presented in this Chapter.

General guidelines

The general approaches to organization design differ w.r.t. the presence and involvement of agents. The design can be performed without having in mind specific agents, the necessary agent profiles are composed at the later design stages based on the considered/designed tasks. Organizational design can also be performed w.r.t. a (partially) known set of agents who will take roles in the organization. Thus agents' skills and traits can be taken into account. Sometimes the agents are not only known but they have some degree of power to steer the design process.

The design process often starts with the identification of one or more high-level goals of an organization. These goals (initially still informally defined) should answer the question: why should the organization exist, what purpose will it serve? Such goals can be identified by the designer or emerge through communication and/or negotiation between the involved agents. In the second case the resulting organizational goals reflect to some extent the individual goals of

the participating agents. In this way some possible conflicts between individual and organizational goals are prevented early. If conflicts appear, they can be addressed through negotiation and redesign at the later stages.

The higher-level goals are often more abstract. Through refinement, more specific, easier to evaluate, goals are formulated. Also, often the higher-level goals are long-term, strategic goals while their sub-goals are shorter-term tactical or operational goals. The leaves of the hierarchies should be goals formulated so that the corresponding PIs can clearly be associated with the processes in the workflow. In this way the satisfaction of every goal in the hierarchies can be evaluated.

Also at the earlier stage of the design process one or more general tasks are identified answering the question: what should the organization do? For identifying these tasks sometimes only the defined goals are considered. However when the involved agents are (partially) known, the definition of tasks can be based on the available skills and experience as well. These tasks are refined into task hierarchies. The used/produced resource types are identified which can also form hierarchies. Based on the tasks, processes are defined and organized into workflows possibly at different levels of abstraction. The level of elaboration of these structures can depend on the type of the organization. In mechanistic organizations the procedures are prescribed to a great degree of detail resulting in elaborate structures refined to simple tasks and processes. In organic organizations (e.g., adhocracies) the procedures are described at a higher-level of abstraction leaving the agents more freedom to choose how to perform them resulting in shallow task hierarchies and less elaborate workflows.

The design process can follow different paths through the views and concepts but several general guidelines can be formulated. When an informally defined goal is formalized and made more precise this should reflect on the PI structure, often meaning to define a new or revise an existing PI. A change in the goal hierarchy should also reflect on the task hierarchy by identifying new or existing tasks that can realize the new or revised goals. A change in the task hierarchy often brings changes to the current workflow design. Adding/revising processes in the workflow might give rise to new PIs to be monitored. When a PI is proposed it should be decided on its level of importance to understand if a new goal should be formulated based on it. The definition of roles is based on the currently defined tasks and processes. Fig. 9 shows the main dependencies between concepts and structures guiding the design process.

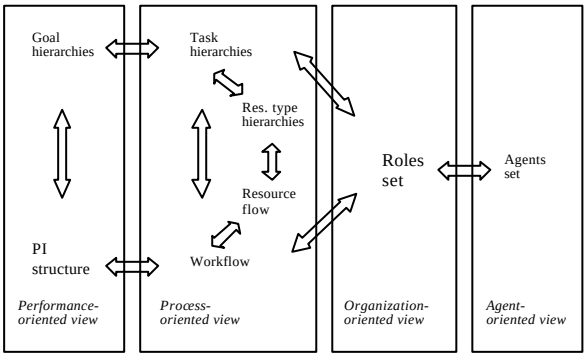


Fig. 9 Dependencies between the structures of the four views

Power and authority relations between the defined roles are usually assigned at the later design stages. However different general schemes can be predefined and committed to at the earlier stages as well leaving the details for later. Such schemes reflect different types of organizations identified in organization theory such as: hierarchical, flat or team-based organizations which differ in the way the power is distributed, granted or accepted by roles (agents).

The choice of scheme should be driven by an analysis of the environment in which the organization should operate. For example a relatively stable environment tolerates a well-defined hierarchical structure which allows more efficiency. A changing environment can be addressed by a lighter, flexible and dynamic structure adaptable to changes. The environment in which the organization will operate should be considered at every design step and every view of the framework.

Sometimes instead of designing an organization from scratch, a specification of an existing one is created. Here a wide range of internal or external documents are used, e.g., company policies, job descriptions, mission statement, procedure descriptions, laws. However even the richest documentation leaves some information unspecified thus it is essential to involve domain experts and managers.

The specification process for the case study

The specification of the organization in the case study is designed based on existing job descriptions for the employees and documents describing the company mission, goals, structure and procedures. The process started with defining the involved roles, their decomposition and the general outline of the authority and power relations in the organization. For each role, its main tasks are defined in the job descriptions. These were used as basis for the construction of the task hierarchies. The predefined tasks were formalized and refined or aggregated where necessary introducing new tasks in the lower or higher levels of the hierarchies. Using the lowest-level tasks and the documentation on company procedures, the processes in the workflow and their relationships were defined. The company procedures on some aspects of the organization are relatively well documented such as the path of a customer contract, however, other are scarcely documented, e.g. training, evaluation and recruitment. There mostly the employees' job descriptions were used. Together with the tasks and processes, the relevant resource types and resources were identified which mostly describe forward, short-term and daily plans and various data coming from company's information systems, data change forms, various reports and summaries for the management team.

In parallel to the specification of the process-oriented view, the performance-oriented view was also specified. Here, the documents on company's mission and general goals were used as well as the job descriptions where some performance measures and goals for the individual roles are defined. Company's mission statement and goals were defined at a very high abstraction level using notions as customer satisfaction, employees' motivation, efficiency, effectiveness of operations. These were used to define the highest-level goals in the goals hierarchies. While the goals were refined, reformulated and formalized, the related PIs were identified. In contrast, the PIs and goals extracted from the job descriptions were very specific, however incomplete. They were used to define lower-level goals. Where necessary, intermediate goals were inserted to complete the goals structures. Other goals were based on statements from various company documents defining performance measures. First the PIs were defined, then the related goals. When the backbones of the process-oriented and the performance-oriented views became available, the process of relating them started by assigning goals and PIs to processes and tasks.

Finally, the specific authority and power relations were defined and formalized based on the roles and tasks hierarchies. Also the input and output ontologies for the roles were defined. The job descriptions provide well-defined requirements for agents performing the roles which were incorporated in the specification as capabilities required for roles.

Domain-specific constraints were extracted from all available documents at various steps of the process. Most of them were imposed by the organization to regulate its internal procedures and ensure high standard of service and communication with customers. Domain-specific constraints imposed by the physical world were mostly missing in the documentation as they were considered obvious. These were added based on domain knowledge.

ANALYSIS OF ORGANIZATIONAL STRUCTURES AND DYNAMICS

The formal foundations of the proposed framework enable three types of automated analysis (Sharpanskykh, 2008): consistency verification of specifications of every view (i.e., establishing the correctness w.r.t. a set of constraints), validation of correct specifications by simulation, analysis of actual executions of organizational scenarios based on specifications. In the following these analysis types are discussed briefly.

In the performance-oriented view a set of constraints and reasoning rules that ensure the consistency of PI and goal structures has been defined (Sharpanskykh, 2008). In the process-oriented view, structural consistency constraints are defined for the three types of structures: workflow, task and resource hierarchies. Verification of these constraints is supported by automatic tools. Workflow specifications can be analyzed at different abstraction levels. The organization-oriented view identifies sets of generic consistency constraints on interaction structures of roles and on formal authority relations. To check such constraints, both the interaction and authority structures are translated into the graph representation, in which each vertex corresponds to a role and each edge corresponds to an interaction/authority relation. Then, using algorithms from the graph theory the satisfaction of constraints can be established. The structures of the organization-oriented view can also be analyzed at different aggregation levels.

Based on correct specifications, simulation can be performed, in which different types of agents, defined using the concepts from the agent-oriented view, are allocated to the organizational roles. By considering different simulation scenarios of organizational behavior, the validation of organizational specifications can be performed using the dedicated tool.

To specify simulation models the temporal language LEADSTO (a sublanguage of the TTL) is used (Sharpanskykh, 2008) which enables modeling direct temporal dependencies between state properties. A specification of dynamic properties in LEADSTO format is executable and can often easily be depicted graphically. The simulation tool generates the simulation results in the form of a trace. Traces can be used for the validation of specifications by checking dynamic TTL properties in the environment TTL Checker (Sharpanskykh, 2008). Given a trace and a formalized property as input, the software will generate a result (positive or negative).

Correct organizational specifications can be used to guide and control the actual execution of processes. The execution data recorded by an EIS and structured in the form of a trace can be checked for conformity to a formal organization (the specification and the set of constraints). For this, the specification of the formal organization is translated into properties expressed in the execution language used for the formalization of the trace. These properties are checked on the trace using the TTL Checker software. Moreover, the designer may specify additional properties to be checked.

The traces may also be used to evaluate the PIs associated with the executed processes. These PIs are related to the leaves of the goals hierarchy, thus the satisfaction of these goals can be evaluated. The satisfaction values are propagated upwards to establish the satisfaction of higher-level goals determining the overall organizational performance.

CONCLUSIONS

This chapter describes a formal framework for modeling and analysis of organizations. The framework includes a rich ontological basis that comprises concepts and relations partitioned into four dedicated views: process-oriented, performance-oriented, organization-oriented and agent-oriented view. The framework can be used for representing different types of organizations ranging from mechanistic to organic.

The framework allows defining different types of constraints using the dedicated formal languages of the views. Moreover, it incorporates agent-based models of individuals based on social theories. In contrast to many existing architectures, the proposed framework allows performing different types of automated analysis of particular views and across views. Organizational specifications can be represented and analyzed at different levels of abstraction, which allows handling high complexity and increases scalability of modeling.

The framework allows reuse in several ways that accelerates and facilitates the modeling process. Libraries of commonly appearing parts of structures (goals and tasks hierarchies, PI-structures, workflow graphs, etc.) can be reused for organizations in similar domains. The used tool allows defining parameterized templates (macros) for TTL formulae that can be instantiated in different ways which can also be used as support for designers not skilled in logics.

The views of the proposed framework are formalized based on intuitive, close to the natural, predicate languages, with concepts and relations that can be represented graphically. Currently, the graphical interface is provided for the performance-oriented view, other views are specified textually using dedicated tools. In the future, modeling related to other views will also be supported graphically which should increase the user-friendliness of the framework.

In the Chapter, the application of the proposed framework has been illustrated by a case study over an organization from the security domain. The proposed framework was also applied in the context of a case study from the area of logistics (http://www.almende.com/deal/). Currently, the framework is used for modeling and analysis of an air-traffic control organization.

FUTURE RESEARCH

Future work is planned to further facilitate the use of the framework in various organizational settings. One direction is to investigate how organizational change can be facilitated and modeled. Organizational change exerts great pressure and risk to an organization and its employees. Often the expected results are not reached, the budget and/or time constraints are exceeded, etc. Techniques to make the process more comprehensive and provide means for the analysis of the new, designed organization before the main investment is made can decrease the risk and provide better chance for success.

Furthermore, templates can be defined for different types of organizations that can be reused and become basis for new specifications. Specific aspects such as hierarchical, flat or team-based structure can be predefined to a certain extent giving the designer the opportunity to customize the template. Case studies to support the usability of the templates should be performed.

The presented framework is designed for modeling a single organization. Its environment can be modeled on a high level as environmental objects and characteristics that can change with time. Modern organizations often enter in cooperation with other organization such as customer or supplier organizations (supply chains) and certain degree of alignment is sometimes necessary. Techniques for modeling and analysis in such setting can be beneficial.

REFERENCES

Burton, R.M., & Obel, B. (2004). *Strategic Organizational Diagnosis and Design: Developing Theory for Application*, Dordrecht: Kluwer Academic Publishers.

Barkaoui, K., & Petrucci L. (1998). Structural analysis of workflow nets with shared resources. In W.M.P. van der Aalst, G. De Michelis, C.A. Ellis (Eds.), *Workflow Management: Net-based Concepts, Models, Techniques and Tools* (pp. 82–95). Lisbon: UNINOVA.

Bernus, P., Nemes, L., & Schmidt, G. (Eds.). (1998). *Handbook on Architectures of Information Systems* Heidelberg: Springer.

Bresciani, P., Giorgini, P., Giunchiglia, F., Mylopoulos, J., Perini, A. (2004). Tropos An Agent-Oriented Software Development Methodology. *Journal of Autonomous Agent and Multi-Agent Systems*, *8*(3): 203-236.
Chapurlat, V., Kamsu-Foguem, B., & Prunet, F. (2006). A formal verification framework and associated tools for enterprise modeling: Application to UEML, *Computers in industry*, *57*, 153-166.
Dalal, N., Kamath, M., Kolarik, W. & Sivaraman, E. (2004). Toward an integrated framework for modeling enterprise processes. *Communications of the ACM*, *47*(3), 83-87.
Dignum, V. (2003). *A model for organizational interaction: based on agents, founded in logic*. Ph.D. Dissertation, Utrecht University.
Ferber, J., & Gutknecht, O. (1998). A meta-model for the analysis and design of organizations in multi-agent systems. In Y. Demazeau (Ed.), *Proceedings of Third International Conference on Multi-Agent Systems* (pp. 128-135). IEEE Computer Society.
Fox, M., Barbuceanu, M., Gruninger, M., & Lin, J. (1997). An Organization Ontology for Enterprise Modelling. In M. Prietula, K. Carley, L. Gasser (Eds.), *Simulating Organizations: Computational Models of Institutions and Groups* (pp. 131-152). Menlo Park CA: AAAI/MIT Press.
Hannoun, M., Boissier, O., Sichman J.S., Sayettat, C. (2000). MOISE: An Organizational Model for Multi-agent Systems. In M.C. Monard, J.S. Sichman (Eds.), *Proceedings of the 7th Ibero-American Conference on AI: Advances in Artificial Intelligence, LNCS 1952* (pp. 156 – 165), Berlin: Springer.
Horling, B, & Lesser, V. (2005). A Survey of multi-agent organizational paradigms. *The Knowledge Engineering Review*, *19*(4), 281-316.
Hubner, J.F., Sichman, J.S., & Boissier, O. (2002). MOISE+: towards a structural, functional and deontic model for MAS organization. In C. Castelfranchi, L. Johnson (Eds), *Proceedings of the 1st International Joint Conference on Autonomous Agents and Multi-Agent Systems* (pp. 501-502), ACM.
Koubarakis, M., & Plexousakis, D. (2002). A formal framework for business process modeling and design. *Information Systems*, *27*(5), 299–319.
Mintzberg, H. (1979). *The Structuring of Organizations*. Prentice Hall, Englewood Cliffs.
Pinder, C.C. (1998). *Work motivation in organizational behavior*. Upper Saddle River, NJ: Prentice-Hall.
De Raad, B., & Perugini, M. (2002). *Big Five Assessment*. Hogrefe & Huber.
Sharpanskykh, A. (2008). *On Computer-Aided Methods for Modeling and Analysis of Organizations*. Ph.D. Dissertation, VU University Amsterdam.
Tham, K.D. (1999). *Representation and Reasoning About Costs Using Enterprise Models and ABC*, PhD Dissertation, Enterprise Integration Laboratory, Department of Mechanical and Industrial Engineering, University of Toronto.
Van der Aalst, W., ter Hofstede, A., Kiepuszewski, B., & Barros, A.P. (2003). Workflow patterns. *Distributed and Parallel Databases, 14*(3), 5–51.
Zambonelli, F., Jennings, N.R., & Wooldridge, M. (2003). Developing multiagent systems: the Gaia Methodology. *ACM Transactions on Software Engineering and Methodology*, *12* (3): 317-370.

ADDITIONAL READINGS

Allen, J.F. (1983). Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26, 832–843.
Bacharach, S.B., & Lawler, E.J. (1980). *Power and politics in organizations*. Jossey-Bass, San Francisco.
Biddle, B. (1979). *Role Theory: Concepts and Research*. Krieger Publishing Co.
Blau, P.M., & Schoenherr, R.A. (1971). *The structure of organizations*. New York London: Basic Books Inc.
Bosse, T., Jonker, C.M., van der Meij, L., Sharpanskykh, A., & Treur, J. (2006). Specification and Verification of Dynamics in Cognitive Agent Models. In C. Butz, N.T. Nguyen, Y. Takama (Eds), *Proceedings of the 6th International Conference on Intelligent Agent Technology*, (pp. 247-254), IEEE Computer Society Press.
Carley, K.M. (1991). A Theory of Group Stability. *American sociological Review*, *56*, 331-354.
CIMOSA – Open System Architecture for CIM; ESPRIT Consortium AMICE (1993). Berlin: Springer-Verlag.

Donaldson, L. (2001). *The Contingency Theory of Organizations*. Sage, London.
Esteva, M., Rodriguez-Aguilar, J.A., Sierra, C., Garcia, P., Arcos, J.L. (2001). On the Formal Specification of Electronic Institutions. In F. Dignum, C. Sierra (Eds.), *Agent-mediated Electronic Commerce: the European AgentLink Perspective* LNAI 1991 (pp. 126-147), Springer.
Ferber, J., Michel, F., Baez-Barranco, J.-A. (2004). AGRE: Integrating Environments with Organizations. In *Proceedings of E4MAS, LNCS, vol. 3374/2005* (pp. 48-56), Springer.
Forrester, J.W. (1961). *Industrial dynamics*, Waltham, MA: Pegasus Communications.
Galbraith, J.R. (1978). *Organization design*. London Amsterdam Sydney: Addison-Wesley Publishing Company.
Giddens, A. (2006). *Sociology*, 5th edition. Polity, Cambridge.
Horenberg, J. (1994). *Organizational Behavior.* Hillsdale, NJ: Lawrence Erlbaum Associates.
Krauth, E., Moonen, H., Popova, V., & Schut, M. (2005). Performance Measurement and Control in Logistics Service Providing. In C.-S. Chen (Ed.), *Proceedings of ICEIS 2005* (pp. 239-247), INSTICC Press.
Lorsch, J.W., & Lawrence, P.R. (1970). *Organization design*. USA: Richard D. Irwin Inc.
Marlow, W.H. (1993). *Mathematics for Operations Research*. New York: Dover.
Maslow, A.H. (1970). *Motivation and Personality*, 2nd. ed. New York: Harper & Row.
Morgan, G. (1996). *Images of organizations*. SAGE Publications, Thousand Oaks London New Delhi.
Omicini, A. (2000). SODA: Societies and infrastructures in the analysis and design of agent-based systems. In M.J. Wooldridge, P. Ciancarini (Eds.), *Proceedings of Agent-Oriented Software Engineering Workshop* (pp. 185–193), Springer.
Pfeffer, J. (1982). *Organizations and organization theory*. Pitman Books Limited, Boston London Melbourne Toronto.
Rao, A.S. (1995). Decision procedures for propositional linear-time belief-desire-intention logics. In *Proceedings IJCAI-95 Workshop on Agent Theories, Architectures, and Languages*, 102-118.
Romme, A.G.L. (2003). Making a difference: Organization as design. *Organization Science, 14*, 558-573.
Sadler, P.J., Webb, T., & Lansley, P. (1974). *Management style and organisation structure in the smaller enterprise : a report on a research project conducted with the financial support of the Social science research council*. Ashridge: Ashridge Management Research Unit.
Scheer, A-W., & Nuettgens, M. (2000). ARIS Architecture and Reference Models for Business Process Management. In W.M.P. van der Aalst, J. Desel, A. Oberweis (Eds.), *Inter-operability of Workflow Applications: Local Criteria for Global Soundness, LNCS 1806* (pp. 366-389). Berlin: Springer.
Scott, W.G., Mitchell, T.R., & Birnbarum, P.H. (1981). *Organization theory: a structural and behavioural analysis*, USA, Illinois: Richard D. Irwin inc.
Scott, W.R. (2001). *Institutions and organizations.* SAGE Publications, Thousand Oaks.
Van der Aalst, W.M.P., & van Hee, K. (2002). *Workflow Management: Models, Methods, and Systems*. MIT Press.
Walter, B. (1968). *Modern systems research for the behavioral scientist*, Chicago: Aldine Publishing Co.
Warner, M., & Witzel, M. (2004). *Managing in Virtual Organizations*. Thomson Learning.
Yilmaz, L. (2006). Validation and Verification of Social Processes within Agent-Based Computational Organization Models. *Computational & Mathematical Organization Theory, 12:*4, 283-312.
Yu, E. (1997). Towards Modelling and Reasoning Support for Early-Phase Requirements Engineering. In Proceedings of the 3rd IEEE International Symposium on Requirements Engineering (pp. 226-235). IEEE Computer Society.
Yukl, G. (2006). *Leadership in organizations.* 6edn. Englewood Cliffs, NJ: Prentice-Hall.

Key terms

formal organizational modeling – organizational modeling using formal methods
formal organizational analysis – organizational analysis using formal methods
agent organizations
human organizations

organizational performance

constraints - an expression over objects and/or processes that limits the organizational behavior

process-oriented modeling – modeling of organizational flows of control, resources