Modeling Organizational Performance Indicators

Viara Popova [a,b] and Alexei Sharpanskykh [b]

[a] De Montfort University, Centre for Manufacturing,
The Gateway, Leicester, LE1 9BH, UK
vpopova@dmu.ac.uk

[b] Vrije Universiteit Amsterdam, Department of Artificial Intelligence,
De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands
sharp@cs.vu.nl

Abstract—Performance measurement and analysis is crucial for steering the organization to realizing its strategic and operational goals. Relevant performance indicators and their relationships to goals and activities need to be determined and analyzed. Current organization modeling approaches do not reflect this in an adequate way. This paper attempts to fill the gap by presenting a framework for modeling performance indicators within a general organization modeling framework.

Keywords— enterprise architectures, information systems, modeling, performance indicators

1. Introduction

Measuring and analyzing organizational performance plays an important role in turning organizational goals to reality. The performance is usually evaluated by estimating the values of qualitative and quantitative performance indicators (e.g., profit, number of clients, costs). It is essential for a company to determine the relevant indicators, how they relate to the formulated company goals and how they depend on the performed activities. Nowadays many managers recognize this and put conscious effort in defining company-specific goals, performance indicators and evaluate them. However in practice such analysis is usually done in an informal, ad-hoc way and will benefit from a more systematic approach. The first step towards an improvement in this area is to make explicit the available knowledge on performance indicators and how they are related. In order to use this knowledge in a modern framework for organization modeling it is necessary to formalize the concept of a performance indicator together with its characteristics, relationships to other performance indicators and relations to other formalized concepts such as goals, processes and roles. This will not only contribute to the design and analysis of organizations and the evaluation of their performance but will also enable reuse, exchange and alignment of knowledge and activities between organizations (for example supply chains). This paper introduces a framework for modeling performance indicators and the relationships between them, which constitutes a part of a general framework for organization modeling and analysis. The contribution of the paper can be summarized in the following points:

(1) formalization of the concept of a performance indicator (PI) which has rarely been done in the literature on enterprise modeling;

(2) formalization of the relationships between performance indicators – an issue that has been overlooked so far in the literature on enterprise modeling;

(3) integration of the concept of a performance indicator in an expressive and intuitive formal modeling framework covering all important aspects of a modern organization – in this way it is possible to model the relations between performance indicators and various other concepts such as goals, processes and roles;

(4) it also proposes a mechanism to evaluate organizational performance using the general framework;

(5) the formal foundations of the framework allow formal analysis and verification;

(6) the paper provides methodological guidelines for practitioners on extracting performance indicators and their relationships and on building the relevant structures of the frameworks.

In the general framework, organizations are considered from different perspectives (views). *Process-oriented view* describes the workflow as well as static structures of tasks and resources. *Performance-oriented view* is characterized by a goal structure, a performance indicators structure, and relations between them as well as relations between goals and tasks, performance indicators and processes, goals and roles or agents. *Organization-oriented view* defines the organizational roles, each associated with a set of tasks and characterized by authority and responsibility relations on tasks, resources and information. Commitment, obligation and power relations and sets of competences required for agent allocation to roles are also defined. *Agent-oriented view* identifies different types of agents with their capabilities and principles for allocating agents to roles based on the matching between agent capabilities and competences required for roles. The general framework is operational and parts of it have already been implemented.

The formal language and axiomatic basis for *modeling performance indicators* within the *performance-oriented view* are described in this paper. Based on PIs, goals are constructed in a stepwise formal way as follows: First, a mathematical or logical expression over a PI(s) is defined, which serves as a basis for a formal goal expression. A PI expression divides the PI's domain into sets of values. The meaning of these sets becomes explicit by defining goal patterns at the second step. A goal pattern defines which set of values of a PI(s) is favorable for the organization and which is unwanted. Furthermore, a goal pattern adds the temporal dimension to a PI expression, defining when the PI(s) should (or should not) have favorable (or unwanted) values (e.g., at some time point, during a time interval). A goal is an organizational objective defined at the last step by adding to a goal pattern information about desirability and priority.

Using explicitly defined relations between PIs and goals the paper describes the performance evaluation process and methodological issues of creating and revising performance-oriented models. Furthermore, some verification techniques specific for performance-oriented organization models are briefly discussed.

The presentation is organized as follows. Section 2 gives an overview of the general organization modeling framework. The case study used for illustration is described in Section 3. In Section 4 the main concepts are defined. The relationships between them are defined in Section 5 as well as the semantic aspects of the introduced language with its axiomatic basis. Section 6 discusses the evaluation of organizational performance. Methodological guidelines are given in Section 7. Section 8 describes the implementation of the environment for modeling and analysis of PI specifications. Section 9 discusses related work on modeling organizations and section 10 – the related work on performance measurement. The evaluation of some aspects of the proposed approach is provided in Section 11. Section 12 concludes the paper with a summary and future research directions.

2. The Proposed Framework

The approach for modeling performance indicators presented in this paper is part of a general framework which consists of a modeling approach, based on a number of formal languages, analysis methods for different aspects of an organization model and a methodology, describing the process of organization design.

In order to reduce the complexity of the modeling process and to capture essential aspects of structure and behavior of organizations of different types, the general framework incorporates a number of interrelated concepts that are grouped under different views. Each view represents a certain perspective on an organization and is formally described by dedicated logic-based languages. The following four views are distinguished in the framework: process-oriented, performance-oriented, organization-oriented and agent-oriented view. Note that both the components of the proposed framework and the considered modeling views can be directly related to the components of the GERAM (the Generalized Enterprise Reference Architecture and Methodology) [28], the initiative of the IFIP-IFAC Task Force, based on which several international standards for enterprise modeling have been created. The GERAM provides a generalized template for the development of elaborated enterprise modeling frameworks, which was taken into account when the proposed framework has been developed. Please note that the GERAM itself does not elaborate enterprise modeling concepts and does not introduce any modeling language, but rather defines a set of structural and behavioral requirements for other elaborated enterprise modeling frameworks. In the following a brief characterization of the listed organization modeling views is given.

Process-oriented view: This view describes different types of flows in an organization, i.e., the flow of processes, resources and information. To define these flows, appropriate static structures of tasks and resources are used. For example, a task structure is a graph built based on the refinement relations between tasks. Processes are instances of tasks in the workflow. The workflow reflects a (partial) temporal ordering of processes, influenced by control elements (i.e., conditions and decision points) that determine the path of the flow. Resources are material or informational products that are produced, used or consumed during the execution of processes. Examples can be: raw materials, components, transporatation vehicles, databases, etc. The process-oriented view is presented in more details in [38].

Performance-oriented view: This view includes an organizational goal structure, a performance indicators structure, and relations between them. Both these structures are built based on multiple types of relations that allow representing to a high degree the complexity and interdependency of objectives and performance measures within real organizations. In order to evaluate and to predict a level of performance of an organization, next to relations between goals and performance indicators, relations between goals and tasks, and performance indicators and tasks are explicitly defined in this view. Furthermore, organization goals are always assigned to certain organization role(s). Also agents may have individual goals that conform to or contravene goals of an organization and, thus, influence the execution of organizational tasks by agents. Both organizational and individual goals and relations between them are specified in this view. This paper introduces the part of the performance-oriented view that is dedicated to modeling performance indicators. The rest of the performance-oriented view describing the approach for modeling organizational goals is presented in [39].

Organization-oriented view: Within this view organizational roles are defined. Each role is associated with a subset of functionalities (a set of tasks) performed by an organization and, consequently, is committed to certain organization goal(s). Each role is characterized by authority and responsibility relations on tasks, resources and information defined in the process-oriented view. Between roles power relations with respect to certain tasks are defined. For each role a set of competences is defined that are required from agents to be allocated to the role. The organization-oriented view is presented in [43].

Agent-oriented view: In this view different types of agents with their capabilities (i.e., knowledge and skills) are identified. Principles for allocating agents to roles are formulated based on the matching between agent capabilities and competences defined for roles. More details on the agent-oriented view can be found in [44].

The identified views are related to each other by means of sets of common concepts. For example, the relations between goals and roles are introduced in the performance-oriented view. Further these relations are used in the organization-oriented view to describe mechanisms of goal assignment and delegation, which also use power and authority relations from the organization-oriented view.

In all these views environmental conditions, in which the organization is functioning, are taken into account: they influence the specification of organization concepts and relations between them (e.g., the formulation of goals and the specification of tasks), thus, affecting the structure and behavior of a particular organization model. Furthermore, the type of the environment determines a part of domain knowledge, which is represented by facts and rules about the environment that directly influence all the activities within the organization. Another part of the domain knowledge is defined by intrinsic properties of the organization itself. Modeling of the environment may vary in different organizational specifications. In particular, in some cases it can be defined by a set of objects with certain properties and states and by causal relations between objects. While 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).

Data about the actual execution of organizational processes are recorded by most workflow management systems. Examples of data that are often registered are: starting and finishing time points of processes, types and amounts of resources used/consumed/produced/broken, names of actors who perform processes, etc. Such data forms actual execution traces which can be analyzed and used to evaluate the organizational performance. Furthermore such traces can be used to detect potential problems, discrepancies from the model, bottlenecks, etc. Details on the mechanisms to analyze execution traces can be found in [38]. Section 8.2 gives more information on how traces in the Temporal Trace Language can be analyzed in the software environment TTL Checker [7].

The identified views with their concepts and relations have some similarities with the constructs introduced by the standard ENV 12204 [22]. These constructs represent templates used for composition of enterprise models. Relationships between constructs, static and behavioral, are described implicitly in this standard. Furthermore, the constructs and relations between them are specified only syntactically, without providing precise semantics.

To formalize concepts, relations and rules specified within the described views, dedicated first-order sorted predicate logic-based modeling languages are used. To express temporal relations in specifications of the views, the dedicated languages of the views are embedded into the Temporal Trace Language (TTL) [7], which is a variant of the order-sorted predicate logic. These languages allow formal representation of both quantitative and qualitative aspects of an organization model, described by continuous and discrete variables respectively.

A formal representation of the organization models enables different types of analysis: analysis by simulation of different scenarios of organizational behavior, and analysis by formal verification and validation techniques for different aspects of an organization structure and behavior. These techniques can be applied for:

(1) analyzing an organization model abstracted from agents (i.e., without allocation of agents to roles) for the purpose of identifying inconsistencies, redundancies and errors;

(2) analyzing a simulated organization model with agents allocated to roles in a certain scenario;

(3) analyzing a model based on empirical data generated by execution of processes in real organizations.

Some of these analysis techniques are specific for a particular view; however, many of them make use of relationships between elements of an organization model defined across different views.

The methodological part of the framework describes a number of guidelines for designing organization models of different types (i.e., corresponding to different views). Design principles defined in these guidelines may be applied both for creating organization models from scratch, and for analyzing models of existing organizations. In order to provide freedom of choice for a designer during the organization design process, most of the design principles have the status of a recommendation, which nevertheless may be very useful for novices and inexperienced users.

This paper describes a modeling approach and a formal language for modeling performance indicators within the performance-oriented view. Other views defined in the proposed framework are considered elsewhere [38, 39, 43].

In the next Section the case study that will be used for illustration of the proposed performance-oriented framework is introduced. In Section 4 the main concepts related to performance indicators are defined. The relationships between them are described and formalized using the dedicated logic-based language in Section 5. Section 6 discusses how the introduced concepts are used to evaluate the performance of the organization. Methodological guidelines concerning the construction and the revision of the PIs structures are given in Section 7.

3. Introduction to the Case Study

The proposed approach is applied for modeling and analyzing an organization from the security domain. The main purpose of the organization is to deliver security services (e.g., private property surveillance, safeguard) to different types of customers (individual, firms and enterprises). The organization has well-defined structure with predefined (to a varying degree) job descriptions for employees. The total number of employees in the organization is approximately 230 000 persons working in several regions. The global management of the organization is performed by the board of directors, which includes among others the directors of the different divisions (corresponding to the different regions). Within each region a number of areas exist controlled by area managers. An area is divided into several units, controlled by unit managers. Each unit serves a number of locations, for which the contracts with customers have been signed and security officers are allocated. The allocation of employees is performed based on plans created by planning groups.

The model that corresponds to the part of the organization concerned with the planning process will be used in this paper to illustrate concepts, relations and techniques related to the performance-oriented view. Therefore, the planning process is described here in more detail. The planning process consists of forward (or long-term) planning and short-term planning. Forward planning is the process of creation, analysis and optimization of forward plans that describe the allocation of security officers within the whole organization for a long term (4 weeks). Forward plans are created based on customer contracts by forward planners from the forward planning group. During the short-term planning, plans that describe the distribution of security officers to locations within a specific area for a short term (a week) are created and updated based on the forward plan and the available up-to-date information about the security employees. Furthermore, based on short term plans, daily plans are created. Within each area short-term planning is performed by the area planning team that consists of planners and is guided by a team leader. During the planning process short-term planners interact actively with forward planners (e.g., for consultations, problem solving). Furthermore, forward planners have a number of supervision functions with respect to short-term planners.

This case study is used for illustration of the concepts and relationships defined in the rest of the paper. Appendix A gives the full details of the specification of the performance indicators for this case study using the definitions of Section 4. The examples in the rest of the paper refer to these performance indicators using the same numeration as in Appendix A.

4. Performance Oriented Concepts

This and the following Sections present the contribution of this paper – the approach for modeling performance indicators within the performance-oriented view and how performance indicators are related to the rest of the general framework.

Every organization exists for the achievement of one or more goals. This varies depending on the type of organization, e.g. the main goal of a manufacturing company can be the realization of maximal profit while the goal of a non-profit organization can be to effectively protect wild animals. Being aware of these goals is a prerequisite to taking measures for their satisfaction. To ensure continued success, the organization should monitor its performance with respect to its goals. The notions of a goal and a performance indicator are therefore essential. They are the main building blocks of the performance-oriented view of our approach. Figure 2 gives a graphical representation of the concepts and relationships of the *performance-oriented view*. This paper describes a modeling approach and a formal language for PIs as part of the performance-oriented view. Other concepts and views are presented in details elsewhere.

In the following Subsection, the definitions for performance indicators and performance indicator expressions are given with their specific characteristics. Subsection 4.2 briefly presents the other notions of the performance-oriented view – goal and goal pattern which are closely related to the notion of a performance indicator. Subsection 4.3 gives a short overview of the main concepts of the other three views of the framework which are necessary for the evaluation of performance.

4.1. *Perfomance Indicators*

A ***Performance indicator*** is a quantitative or qualitative indicator that reflects the state/progress of the company, unit or individual. The following characteristics can be specified for each performance indicator:

Name;

Definition;

Type – continuous or discrete, for example if the indicator can be measured as a continuous number then it's type can be specified as continuous and if the indicator is measured in indivisible units such as packets, items, pieces or using predefined concepts such as low/medium/high then it can be specified as discrete;

Time frame – (if applicable) for which time frame is the performance indicator defined, the length of the time interval for which it will be evaluated, e.g. the indicator ‘yearly profit’ has time frame ‘year’, ‘number of customers per day’ has time frame ‘day’;

Scale – if relevant, the measurement scale for the performance indicator, different scales can be predefined and referred to here;

Min value, Max value – when a predefined scale is used and only a part of this scale is relevant for the particular performance indicator;

Source – which was the internal or external source used to extract the performance indicator: company policies, mission statements, business plan, job descriptions, laws, domain knowledge, etc. – these sources contain (informal) statements about the desired state or behavior of the company and regulations it has to obey;

Owner – the performance of which role or agent does it measure/describe;

Threshold – the cut-off value separating changes in the value of the performance indicator considered small and changes considered big; used to define the degree of influence between performance indicators (see Section 5); depending on the scale of measurement of the performance indication, the threshold can have a clearly-defined unit of measurement (e.g., measured in hours, km, number of persons or products, etc.) or it can be measured in unnamed units for qualitative scales such as low-medium-high where one unit represents the difference between two consequtive points on the scale, for example between low and medium or between medium and high (which are assumed to be equidistant) – in the example below PI5 has a threshold of 2 units which is the distance between very_low and medium, between low and high and between medium and very high;

Hardness – a performance indicator can be soft or hard where soft means not directly measurable, qualitative, e.g. customer’s satisfaction, company’s reputation, employees’ motivation, and hard means measurable, quantitative, e.g., number of customers, time to produce a plan.

Example:

PI name: **PI5**;

Definition: average correctness of produced plans

Type: discrete;

Time frame: month;

Scale: very_low-low-med-high-very_high;

Source: mission statement, job descriptions;

Owner: forward and daily planning departments

Threshold: 2 units;

Hardness: soft;

PI name: **PI27**;

Definition: time to create a new short-term plan after all operational data is received

Type: continuous;

Time frame: month

Scale: REAL

Min value: 0;

Max value: max_time_CSP;

Unit: hour;

Source: job descriptions

Owner: daily planning departments

Threshold: 24 hours;

Hardness: hard;

PI name: **PI29**;

Definition: efficiency of allocation of security officers to objects;

Type: discrete;

Time frame: month

Scale: very_low-low-med-high-very_high;

Source: job descriptions

Owner: forward and daily planning departments

Threshold: 24 hours;

Hardness: soft;

Appendix A contains the list of performance indicators identified for the case study for the forward and daily planning departments with the corresponding characteristics. The list contains a lot of indicators that are often used in practice and demonstrates how they can be represented in the framework. Also less conventional indicators can be represented in a similar way as long as the necessary information is available.

The process of discovering performance indicators is not trivial. It is company-specific and can use information and expertise from different sources. The relevant performance indicators for a specific organization can sometimes be found in documents, either internal or external for the organization, such as company policies, mission statements, business plan, job descriptions, laws, domain knowledge, etc. however often this information is incomplete and imprecise. Here the expertise of domain experts, managers and other involved parties should be used. Sometimes important indicators remain implicit as they are considered obvious. Conscious effort is needed to make such information explicit so that it can be incorporated in the design process.

Here existing research can be used as well where performance measurement systems and key performance indicators are defined for specific domains and investigated with respect to their usefulness (e.g., [11, 14, 33, 47]). Also partial reuse of previous models of organizations in a similar domain can be possible,

Formalizing performance indicators includes the definition of all relevant characteristics which makes the implicit knowledge explicit. The process of extracting the performance indicators from source documents involves asking the question: What should be measured / observed to ensure the requirements in the document? Performance indicators are often represented by nouns in the text; modifiers such as adjectives give information about the type, scale of measurement and what is considered a desirable value of the performance indicator (used in performance indicator expressions, goal patterns and goals), e.g., the job description requires a planner 'to ensure high accuracy of calculation when creating a plan' then 'accuracy of calculation in plan creation' is a performance indicator and 'high' – its desired value.

Sometimes such a process is straightforward, for example it is clear how profit can be defined and measured. However in other cases important and sometimes subjective choices need to be made. Often the performance indicators that can be extracted from documents such as the mission statements and policies are soft and difficult to assess. In order to evaluate such a performance indicator it is usually beneficial to find one or more closely related hard indicators that can be measured instead and that can give an impression on the state of the soft one. For example customer satisfaction is often an important indicator which is very difficult to assess. One way to address the problem is to represent it as a soft indicator with an appropriate scale (e.g., very_low-low-medium-high-very_high) and depend on a (to a large extent) subjective evaluation. It is also possible to make an additional assumption that it can be measured sufficiently correctly by carefully designed questionnaires or by a combination of other (easier to measure) indicators such as the percentage of returning customers, the percentage of on-time deliveries, the number of complaints, etc. Which of these approaches and measures will be most relevant is company-specific and should be decided involving the management and domain experts.

Figure 1 shows the sources of information used for extracting performance indicators and the main stages of the process.

Figure 1: The process of extracting performance indicators

The set of performance indicators that can be defined for one organization can be very large and it is often not feasible to monitor all of them. Therefore the companies select a subset of indicators, called *key performance indicators*, that can give a representative picture of the performance and the costs of measuring and monitoring are reasonable. It is essential for the company to choose its key performance indicators carefully to form a balanced (with respect to the company activities, involved parties, etc.) and sufficiently complete set [32]. The key performance indicators of the organization should be reflected in its goals. The choice of key performance indicators is strongly company-specific and depends on the state, orientation and positioning of the organization on the market, its mission statement and vision. These will define a preference for some performance indicators as more important than others. For example, cost efficiency and profit might be central for one organization, others might include customer satisfaction (e.g., if the success of its operations depends strongly on returning customers), employees' motivation (e.g., if sufficiently qualified personnel is difficult to recruit), environmental impact, growth, market share, etc.

The second concept relevant for modeling performance indicators is the performance indicator expression which is defined below.

Performance indicator expression – a performance indicator or a mathematical statement over a performance indicator containing >, ≥, =, < or ≤. A performance indicator expression can be evaluated to a numerical, qualitative or Boolean value for a time point, for the organization, unit or agent. For example using the above defined performance indicators we can formulate performance indicator expressions as follows: PI27 ≤ 48h; PI5 = high.

4.2. *Goals*

Within the performance-oriented view also goal patterns and goals are defined which are addressed in detail in [39]. In the following paragraph a brief overview is given. By establishing the link between performance indicators and goals, mechanisms for evaluating the performance are made possible by means of measuring performance indicators, using the values to assess the satisfaction of low-level goals and propagating to establish the satisfaction of high-level goals.

Performance indicator expressions are used to define goal patterns which are properties that can be checked to be true or false for the organization, unit or individual at a certain time point or period. Three types of goal patterns have been introduced: (1) achieved (ceased) – it should be checked whether the property is true (false) for a specific time point; (2) maintained (avoided) – it should be checked whether the property is true (false) for the duration of a specific time interval; (3) optimized (maximized, minimized, approximated) – it should be checked if the value of the PI expression has increased, decreased or approached a given target value for the duration of a given time interval. For example a goal pattern of type *achieved* based on PI27 can be GP1:'achieved that PI27 ≤ 48h'. Goals are objectives that describe a desired state or development and are defined by adding to goal patterns information such as desirability and priority. A goal based on GP1 can be G1:'It is required that the state is achieved in which PI27 ≤ 48h'. This goal is an example of the *achievement goal* type that is formulated using the *achived* goal pattern. Another goal type - *development goal* - is based on goal pattern of type *maintained, avoided* or *optimized* (*development goal*). For example, development goal G2: '*It is desired that the state is maintained, in which PI29 = high*' is formulated based on the goal pattern 'PI29 = high' using the goal pattern *maintained*.

Goals can be hard (satisfaction can be clearly established) or soft (satisfaction cannot be clearly established). Satisfaction of a hard goal can be established quantitatively. Hard goals have labels ordered as follows: *satisfied* > *undetermined* > *failed.* For soft goals degrees of *satisficing* are defined. Soft goals are given labels that correspond to their degrees of satisficing/denial with a natural order between the labels: *satisficed* > *weakly_satisficed* > *undetermined* > *weakly_denied* > *denied.*

Goals can be organizational (i.e., belong to an organization, unit or role) or individual (i.e., belong to an agent). Individual goals may comply with, be disjoint or conflict with organizational goals. This can be determined by analyzing the relations between the performance indicators used in the definitions of the goals (see Section 7).

Goal can be refined into subgoals forming a *goals hierarchy*. Hard goals can be refined by, respectively, AND and OR relations. A goal that is refined into an AND-relation is satisfied if and only if all the goals in the list are satisfied. If several refinements are defined, they are considered as alternatives connected by OR, i.e. they allow multiple possibilities to satisfy the goal.

The process of refinement of soft goals distinguishes two types or relations: *satisfices* and *contributes_to*. The *satisfices* relation indicates a strong contribution in a positive way of a goal to the satisficing of another goal. If the first goal is satisfi(c)ed and any other influences are ignored then the second goal is considered satisficed. The *contributes_to* relation indicates a positive contribution of a goal to the satisficing of another goal, however might not be enough to satisfice it.

Information about the satisfaction of lower level goals can be propagated to determine satisfaction of high level goals (see Section 6 which discusses the evaluation of organizational performance).

4.3. *Other Views of the Framework*

The performance-oriented concepts are related to *other views* in the following way. Goals are realized by performing organizational functions described by *tasks*. *Processes* are specific instances of tasks temporally ordered in a workflow and assigned to roles. Performance indicators are associated to specific aspects of the execution of particular processes. In contrast to the i* approach that is often used for goal-oriented modeling [50], in the proposed framework the goals hierarchy is decoupled from the tasks hierarchy and clear relationships between two goals and tasks are defined.

A *role* represents a predefined set of functionalities performed within the organization which can be allocated to *agents*. Roles and agents can be committed to organizational or individual goals respectively. Roles are characterized by sets of *competences*, required to perform a certain task. Competences can be knowledge (facts and procedures, of which the agent has confident understanding) and skills (i.e., developed abilities of agents to use effectively and readily their knowledge for the performance of tasks). Skills are formulated as performance indicator expressions over individual performance indicators. Agents are autonomous entities, characterized by their individual goals and capabilities. Individual goals of agents are based on individual performance indicators. *Capabilities* can be knowledge and skills that are possessed by agents. Skills are formulated as performance indicator expressions over individual performance indicators. An agent can only play a role if it has the capabilities to match the competences required for the role.

By integrating the performance-oriented view in the general framework, the evaluation of performance can be achieved by monitoring processes, measuring the associated performance indicators and assessing the related goals. Furthermore, the performance indicators and goals can be linked to roles and agents to describe individual performance as well as ownership of goals defining individual or group interests.

5. Modeling Relationships on Performance Indicators

The formal language used for specifying the meta-model for the performance-oriented view is a variant of the first order sorted predicate language. In this language, for each concept a special sort is introduced, containing all the names of concept instances (e.g., sort PI contains all names of performance indicators). The characteristics (attributes) of the concepts are represented by relations (predicates) with arguments: a concept name, an attribute name and a value the attribute (e.g., ***has_attribute_value***: PI × ATTRIBUTE × VALUE). In the following for readability such predicates are used in the more compact form: concept.attribute=value. Specific values that have been measured for performance indicators during or after the execution of organizational processes can be specified using the predicate ***PI_has_value***: PI × PI_VALUE where the sort PI_VALUE includes the sort VALUE together with all possible evaluations of soft performance indicators (e.g. low, medium, high). For more details on recording and analyzing the execution of organizational processes and on evaluating performance indicators the reader is referred to [38]. In the following paragraphs, the relations between performance indicators are defined. In order to provide formal meaning and to enable formal verification (e.g., consistency or integrity checking), the axiomatic basis is also defined.

causing: PI × PI × {very_pos, pos, neg, very_neg}: The first performance indicator causes change in the same direction (positive) or opposite direction (negative) to the second performance indicator. Very_positive describes the situation when small change in one performance indicator causes big change in the other. Similarly for very_negative. The distinction between small and big change can be subjective and therefore should be defined carefully by the designer using input from domain experts. It is specific for each performance indicator and is specified in the model by the threshold values assigned to performance indicators. When the value of a performance indicator increases or decreases, positive or negative difference can be calculated and compared to the threshold value to determine whether it is considered a small or big change. This informal explanation of the causality relation can be formalized as follows using the Temporal Trace Language [45] (p1 and p2 are variables over sort PI):

causing(p1, p2, pos) iff:

$\forall\gamma\ \forall t\ \forall a,b{:}\mathrm{PI\_VALUE}\ \mathrm{state}(\gamma,t) |= [\ \mathrm{PI\_has\_value}(p1,\ a) \land \mathrm{PI\_has\_value}(p2,\ b)\ ] \Rightarrow$

$\forall t1{>}t\ [\forall c{:}\mathrm{PI\_VALUE}\ \mathrm{state}(\gamma,t1) |= [\mathrm{PI\_has\_value}(p1,\ c) \land c{>}a\ ] \Rightarrow$

$\exists t2\geq t1\ \exists d\text{:PI\_VALUE state}(\gamma,t2)|= [\text{PI\_has\_value}(p2, d) \wedge d>b]]\ \&\ [\forall e\text{:PI\_VALUE}$

$\text{state}(\gamma,t1)|= [\text{PI\_has\_value}(p1, e) \wedge e<a\ ] \Rightarrow$

$\exists\ t2\geq t1\ \exists f\text{:PI\_VALUE state}(\gamma,t2)|= [\text{PI\_has\_value}(p2, f) \wedge f<b]]$

causing(p1, p2, very_positive) iff:

$\forall\gamma\ \forall t\ \forall a,b\text{:PI\_VALUE state}(\gamma,t)|= [\text{PI\_has\_value}(p1, a) \wedge \text{PI\_has\_value}(p2, b)\ ] \Rightarrow$

$\forall t1>t\ [\forall c\text{:PI\_VALUE state}(\gamma,t1)|= [\text{PI\_has\_value}(p1, c) \wedge c>a \wedge c\text{-}a<p1.\text{threshold}] \Rightarrow$

$\exists t2\geq t1\ \exists d\text{:PI\_VALUE state}(\gamma,t2)|= [\text{PI\_has\_value}(p2, d) \wedge d>b \wedge d\text{-}b>p2.\text{threshold}]]\ \&$

$[\forall e\text{:PI\_VALUE state}(\gamma,t1)|= [\text{PI\_has\_value}(p1, e) \wedge e<a \wedge a\text{-}e<p1.\text{threshold}] \Rightarrow$

$\exists t2\geq t1\ \ \exists f\text{:PI\_VALUE state}(\gamma,t2)|= [\text{PI\_has\_value}(p2, f) \wedge f<b \wedge b\text{-}f >p2.\text{threshold}]]$

The causality relations for the negative and very_negative cases are defined in a similar manner.

correlated: PI × PI × {pos, neg}: The first performance indicator is correlated positively or negatively to the second performance indicator, i.e., changes in the first performance indicator result in changes in the second one in the same (pos) or opposite (neg) direction and the other way round. This is defined by the following axiom:

correlated (p2, p1, pn), where pn:{pos, neg} iff:
causing(p1, p2, pn) & causing(p2, p1, pn)

aggregation_of: PI × PI: The first performance indicator is an aggregation of the second performance indicator meaning that they are the same measure at a different aggregation level. For example ‘attrition per year’ is an aggregation of ‘attrition per month’. If the aggregation relation exists between performance indicators, then these performance indicators are also positively correlated with each other.

$\forall$ p1,p2:PI: aggregation_of(p1, p2) $\Rightarrow$ correlated (p1, p2, pos)

Both performance indicators in the aggregation relation have the same type and unit. This is expressed by the following axiom:

$\forall$ p1,p2:PI: aggregation_of(p1, p2) $\Rightarrow$ p1.type=p2.type & p1.unit=p2.unit

The aggregation relation exists for example between performance indicators of the same type with time frame attributes related by the aggregation relation, e.g., performance indicator 'revenue for a year' is an aggregation for performance indicator 'revenue for a month'. Aggregation relation between performance indicators can be defined based on the relations of performance indicators to processes and organizational roles. More specifically, the performance indicators of the same type are related by aggregation, when their owners (roles, agents) are related by the structural aggregation relation is_part_of, e.g., is_part_of(group1,deptA). For example performance indicator 'number of planners in deptA' is an aggregation of performance indicator 'number of planners in group1'. Similarly if performance indicators of the same type measure the same aspect of execution of process instances of tasks related by is_subtask_of relation, e.g., is_subtask_of(collect_data, create_plan) then often aggregation relation exists between these performance indicators.

Using the standard procedure from the sorted first-order predicate logic, terms and formulae over sort PI can be built, expressing different types of mathematical relations between performance indicators. For example, organizational_profit=organizational_revenue-organizational_costs; (PI1>3 & PI2=4.5) $\Rightarrow$ PI3 > 5.2.

Note that relations between performance indicators may change over time. To express the temporal aspect of relations between performance indicators, Temporal Trace Language is used [45]. For example, the following formula expresses that in a particular organizational scenario described by the trace γ1 (i.e., a temporally ordered sequence of states) performance indicator p1 causes a positive change in p2 during the time interval [t1, t2] and p1 causes a negative change in the value of p2 during the time interval (t2, t3]: $\forall t', t''\ t' \geq t1\ \&\ t' \leq t2 \Rightarrow \mathrm{state}(\gamma 1, t') \models \mathrm{causing}(p1, p2, pos)\ \&\ t'' > t2\ \&\ t'' \leq t3\ \&\ \mathrm{state}(\gamma 1, t'') \models \mathrm{causing}(p1, p2, neg)$

In the following more detailed examples are given in the frames of the case study using some of the performance indicators defined in Appendix A.

Examples:

Name: **PI1**

Definition: The level of correctness of plans with respect to the contracts of the employees, the laws, the general policy of the company and division

Type: discrete

Time frame: month

Scale: very_low-low-medium-high-very_high

Source: mission statement, job descriptions

Owner: forward and daily planning departments

Threshold: 2 units

Hardness: soft

Name: **PI2**

Definition: the level of knowledge of employees involved in (forward) planning about the current contracts of the employees, the laws, the general policy of the company and division

Type: discrete

Time frame: month

Scale: very_low-low-medium-high-very_high

Source: mission statement, job descriptions

Owner: forward and daily planning departments

Threshold: 2 units

Hardness: soft

For these two performance indicators the following relation was discovered:

causing(PI2, PI1, pos)

Name: **PI30**

Definition: average level of optimality of forward, short-term and daily planning for efficient allocation of security officers

Type: discrete

Time frame: month

Scale: very_low-low-medium-high-very_high

Source: job descriptions

Owner: forward and daily planning departments

Threshold: 2 units

Hardness: soft

Name: **PI31**

Definition: average level of optimality of every forward plan for efficient allocation of security officers

Type: discrete

Time frame: month

Scale: very_low-low-medium-high-very_high

Source: job descriptions

Owner: forward planning department

Threshold: 2 units

Hardness: soft

For these two performance indicators the following relation was discovered:

aggregation_of(PI30, PI31)

Figure 3 shows the structure containing the main relationships between the performance indicators that were identified for the case study.

Performance indicators relate to goals, tasks, processes, roles, agents by the following relations:

is_defined_over: PI_EXPRESSION × PI: A performance indicator expression is defined over a performance indicator (i.e., the performance indicator takes part in the expression).

is_based_on: GOAL_PATTERN × PI: The goal pattern in the first argument is defined over the PI in the second argument.

uses: GOAL_PATTERN × PI_EXPRESSION: Goal pattern defined over PI expression.

has_owner: PI × {ROLE, AGENT} × {very_low, low, medium, high, very_high}: A performance indicator measures/describes the performance of a role or agent to the specified degree. Roles can be atomic or composite at any level including the level of the organization.

measures: PI × PROCESS: A performance indicator expresses an aspect of the performance of the process execution, e.g. ‘time to produce a daily plan’ measures the time performance of the execution of the process ‘produce a daily plan’, ‘production costs’ measures the cost performance of the general process ‘production’.

Environmental conditions influence the execution of processes of an organization, thereby, also influence values of performance indicators related to these processes. This influence can be positive or negative and is specified by the following relation:

env_influence_on: ENV_CHARACTERISTIC × PI × {pos, neg}: An environmental characteristic of the sort ENV_CHARACTERISTIC influences a performance indicator in a positive or negative way (i.e., contributes to the increase/decrease of a performance indicator). For example, a large amount of rain contributes negatively to the amount and quality of harvest.

Other types of relations between performance indicators, processes and roles, related to power, supervision, authorization, etc. are discussed in the organization-oriented view [31].

6. Performance Evaluation

Every task in an organization contributes to the satisfaction of one or more organizational goals through performing its process instances. Each goal is formed based on a certain performance indicator(s) which can be measured (directly or indirectly) during or after the process execution depending on the goal evaluation type – in the end or during a certain period of time (evaluation period defined as goal horizon). Data about the actual execution of the processes is recorded by the workflow management system in the form of a trace which can be used for analysis. The satisfaction (degree of satisficing) of the goal(s) is determined by comparing the measured value(s) with the corresponding goal expression(s). Further, the obtained goal satisfaction (satisficing) measure is propagated by applying the rules defined in [39, 40], upwards in the goal hierarchy for determining the satisfaction (degree of satisficing) of higher level goals. In particular, the *satisfied* label is associated with a hard goal refined into an and-list if and only if all goals in this list are satisfied. The propagation mechanisms defined for soft goals are more elaborated and are described in more detail in [39, 40].

Thus, the organizational performance is evaluated by determining the satisfaction (degree of satisficing) of key organizational goals. In the following an algorithm for determining the satisfaction (degree of satisficing) of a goal is provided.

Algorithm: PERFORMANCE-EVALUATION

Input: A key organizational goal g, which satisfaction (satisficing degree) needs to be determined

Output: The satisfaction (satisficing degree) of g

1 Goal g is refined using the goal hierarchy up to the lowest aggregation level; in such a way a set of the lowest levels goals GLL is obtained for g.

2 For each goal gl_i from GLL steps 3-5 are performed

3 For the goal pattern gp_i over which gl_i is formulated a set of PIs is identified on which gp_i is based: PGL_{gli}={ pi| is_based_on(gp_i, pi) }.

4 For each performance indicator pi_i from PGL_{gli} a value from the actual execution trace is determined (the values of the PIs are recorded in an execution trace using predicate has_value: PI x VALUE). If some value(s) of PIs are not recorded in the trace, the satisfaction of the goal cannot be established completely and reliably; the satisfaction (satisficing degree) of g is unknown; **exit**.

5 The degree of satisfaction (satisficing) of gl_i is determined by evaluating the goal expression with the PI values from PGL_{gli}

6 Using the propagation mechanisms from [39, 40] the satisfaction labels of the lower level goals GLL are propagated upwards and the satisfaction (satisficing degree) of goal g is determined.

As an illustration of the proposed performance evaluation procedure consider the following example. For estimating the performance of the organization from the case study, the satisfaction of the key soft goal G3.1: ‘It is required to maintain high efficiency of the planning process’ has to be determined. One of the goals in its refinement is hard goal G3.1.1 ‘It is required to achieve that the number of times the planning activities (creating and updating of a plan) exceed the allowed durations is equal to 0'. This goal is refined into an AND-list that comprises four hard goals: G3.1.1.1: 'It is required to achieve that the time to update a short-term plan given operational data is at most 48 hours', G3.1.1.2: 'It is required to achieve that the time to create a daily plan given operational data is at most 24 hours', G3.1.1.3: 'It is required to achieve that the time to create a short-term plan after all operational data is received is at most a week' and G3.1.1.4: 'It is required to achieve that the time to create a forward plan after all operational data is received is at most a week'. These goals are related to tasks: G3.1.1.1 is realized by the task 'update_shortterm_plan', G3.1.1.2 is realized by the task 'create_daily_plan', G3.1.1.3 by 'create_shortterm_plan' and G3.1.1.4 by 'create_forward_plan'. By measuring the actual execution of the process instances of these tasks, it is determined that the values of the related performance indicators of these goals (PI22: 'time to update short term plan', PI23: 'time to create daily plan', PI27: 'time to create short term plan', and PI28: 'time to create forward plan') do not exceed the prescribed durations. Thus, goals G3.1.1.1, G3.1.1.2, G3.1.1.3 and G3.1.1.4 are satisfied. Due to the AND-refinement relation G3.1.1 is also satisfied and contributes positively to the satisfaction of G3.1 and thus to the overall performance evaluation.

7. Methodological Issues in the Construction of PI Structures

Methodological issues discussed in this Section concern the construction and the revision of the PIs structures. As it was discussed in Section 4, organization's performance indicators can be extracted from different sources

To build a structure of performance indicators, relations between them should be identified using some or all of the following techniques:

(1) Original company documents can be analyzed for finding explicit references to such relations. These can be expressed as statements reflecting the knowledge about the causal relationships

between particular couples of PIs (e.g. “delays in deliveries cause decrease in customer satisfaction”). In some cases PIs can even be defined as formulas based on other PIs and such formulas can be used to extract relationships (e.g. costs for product1 = material costs + production costs + transportation costs + storage costs, therefore changes in the material costs, production costs, transportation costs or storage costs will cause a change in the overall costs for product1).

(2) Knowledge of domain experts and existing libraries of relations between performance indicators may be used. Often company documents do not contain explicit information that can be used for extracting relationships however domain experts can be very useful both due to their knowledge of the literature and their practical experience. Interviews with such experts can be conducted by (or including) the modeler(s). Furthermore, a number of empirical studies can be found in the literature studying the relationships between PIs (e.g. [21] includes an overview of such research on causal or correlational links between PIs). Finally, previously developed models can sometimes be partially reused in the new model.

(3) Performance indicators attributes and relations between these attributes (e.g., relations between time-related attributes and attributes that relate performance indicators to the organization and task structures) can be exploited (see Section 5).

(4) From the existing relations in the performance indicators structure new relations may be inferred (see the rest of this Section).

(5) Data mining techniques may be applied to the data collected during the organization operation.

(6) Intuitions of the modeler may be used after testing by domain experts or simulations.

(7) Relations to the task structure and the goal structure may be exploited (see the rest of this Section).

As it follows from the definitions in Section 5 all the considered types of relations between performance indicators can be reduced to causality relations. Technique (4), discussed in the previous paragraph, allows inference of some missing causality relations from the existing performance indicators structure. Based on the formal definition for a causality relation provided in Section 5, the inference rules are specified in the form causing(p1, p2, s1) & causing(p2, p3, s2) ⇒ causing(p1, p3, s3), where p1, p2 belong to the sort PI and s1, s2, s3 are of sort SIGN={very_neg, neg, pos, very_pos}. More specific (instantiated) inference rules are generated based on Table 1, in which s3 values are given in the cells on the intersection of columns containing s1 values with rows containing s2 values. These inference rules can also be used for verification of the integrity (consistency) of the performance indicators structure.

Table 1. Inference Rules for Causal Relationships

Examples:

Name: **PI1 –** as defined earlier

Name: **PI10**

Definition: level of correctness of administrative processing of all planning data in the system.

Type: discrete

Time frame: month

Scale: very_low-low-medium-high-very_high

Source: job descriptions

Owner: forward and daily planning departments

Threshold: 2 units

Hardness: soft

Name: **PI12**

Definition: level of correctness of administrative processing of short-term and daily planning data in the system.

Type: discrete

Time frame: month

Scale: very_low-low-medium-high-very_high

Source: mission statement, job descriptions

Owner: daily planning departments

Threshold: 2 units

Hardness: soft

For these performance indicators the following relations were identified:

(1) causing(PI10, PI1, very_pos)

(2) aggregation_of(PI10, PI12)

Given the formal definition of an aggregation relation (Section 5), from the second relation follows that:

(3) causing(PI12, PI10, pos)

Therefore based on relations (1) and (3) and the rules in Table 1 we can conclude that:

(4) causing(PI12, PI1, very_pos)

Such inferred relations are not shown on Figure 3 in order to simplify the picture.

Further let us consider technique (7) as earlier defined in this Section. The task structure of an organization may provide insight to discover relations between performance indicators. Often refinement relations specified in the task structure correspond to causality relations in the performance indicators structure. For example, based on the refinement relation between the task ‘create a correct plan’ (related by its process instance to performance indicator ‘time for creating a correct plan’) and its subtask ‘check a plan’ (related to performance indicator ‘time to check a plan’), the performance indicators ‘time to check a plan’ and ‘time for creating a correct plan’ are related by positive causing relation. Refinement relations might also be reflected by other types of relations in the performance indicators structure.

Further, as it follows from the goal definition, goals and performance indicators form two highly interrelated structures – changes in one structure almost always imply changes in the other structure. Thus, the performance indicators structure and the goal structure should be created simultaneously. One possible way is to start by selecting/extracting from the company documentation an objective from which a goal can be formulated. Such objectives are often vague, abstract statements and describe what the company wants to achieve. In the design process it needs to be disambiguated and made more precise while the format of the goal concept is imposed. Here the corresponding performance indicator should crystallize as well as the relevant characteristics of the goal. It is also possible to start with a performance indicator that appears to be important. Such indicators will often be well known measures that are usually relevant for companies, such as profit, revenue, sales, costs, market share, etc. While defining the relevant characteristics of the performance indicator, an important step is also to make explicit what the desirable values of this indicator are. For example high values of profit will be preferred over low values. In this way the first step is taken towards defining the related goal. The next step would be to formalize in a PI expression and a goal pattern. Finally a goal is defined by adding information about priority, desirability and other relevant goal characteristics.

Usually, high level goals of a company are of a strategic (long-term) type. Such goals are often made operational by refining them into lower level tactical (short-term) goals. The identified in such a way refinement relation, by analogy with the task refinement, can be reflected in the performance indicators structure by the corresponding relation between performance indicators, on which the considered goals are based. More specifically, if goals are related by refinement relation, then the corresponding performance indicators are related by a causality relation. Furthermore, if the performance indicator expressions for goals related by refinement, contain comparison functions (e.g. >,<) or measures of degrees (such as ‘high’, ‘low’), or goal patterns are specified by functions such as ‘increased’/‘decreased’, then the specific type of causality may be determined (at least if it is positive or negative). For example, in the case study both goal expressions for G3.1: “It is required to maintain high efficiency of the planning process” and for G3.1.2: “It is required to maintain high level of promptness of communication of forward, short-term and daily planning data to all concerned employees” contain the equality relation to the value “high”. According to the principles explained above, this corresponds to the positive causality relation between the PIs “efficiency of the planning process” and “level of promptness of communication of forward, short-term and daily planning data to all concerned employees”, which is indeed the case in the PI structure.

In general, the refinement and aggregation of goals can be performed based on information about relations in an organization structure, task structure, temporal dependencies and relations between performance indicators.

The identification of conflict relations between goals is of particular importance for the design and evaluation of organizations. To identify such conflicts, the goal patterns and the performance indicators structure can be used. More specifically, by knowing the type of the causality relation between performance indicators and the types of the goal patterns, the presence of a conflict between goals can be determined. For example, the goal ‘It is required to maximize the time for checking the proposed plan for accuracy’ and the goal ‘It is required to minimize the time for producing a correct plan’ are in conflict, since the performance indicators ‘time for examining the plan for accuracy’ and ‘time for producing an accurate plan’ are related by positive causality relation and the corresponding goal patterns are based on opposite types: maximize and minimize.

If a conflict between high level goals is found, then via the refinement the cause of the conflict can be found at the lowest level of the goal structure. For this the relations between performance indicators and the domain knowledge are exploited. Goal patterns of types ‘increase’ and ‘decrease’ are treated in a similar way. Note that this type of analysis is semi-automated, since the approach allows defining a large diversity of goals. The designer should revise every identified possible conflict.

8. Implementation

The LEADSTO modeling environment used to define PI specifications is considered in Section 8.1, whereas the analysis of such specifications using the TTL Checker environment is described in Section 8.2.

8.1 Leadsto Language and LEADSTO Property Editor tool

Predicate logic-based PI specifications can be defined using the LEADSTO language and the LEADSTO Property Editor tool [8]. The LEADSTO language is a sublanguage of TTL that enables modeling direct temporal (causal) dependencies between state properties in successive states. The LEADSTO language allows specifying all structural aspects of the performance-oriented view (i.e., the dedicated language of the performance-oriented view is a sublanguage of LEADSTO). The notion of state as used in LEADSTO is characterized by a set of properties that do or do not hold at a certain point in time. For a given (order-sorted predicate logic) ontology Ont, the propositional language signature consisting of all *state ground atoms* (or *atomic state properties*) based on Ont is denoted by APROP(Ont). The *state properties* based on a certain ontology Ont are formalized by the propositions that can be made (using conjunction, negation, disjunction, implication) from the ground atoms. A *state* S is an indication of which atomic state properties are true and which are false, i.e., a mapping S: APROP(Ont) → {true, false}.

Properties specified in LEADSTO can be executed and can often easily be depicted graphically. The format is defined as follows. Let α and β be state properties of the form ‘conjunction of atoms or negations of atoms’, and e, f, g, h non-negative real numbers. In the LEADSTO language the notation $\alpha \twoheadrightarrow_{e, f, g, h} \beta$ (also see Figure 4), means:

If state property α *holds for a certain time interval with duration g, then after some delay (between e and f) state property* β *will hold for a certain time interval of length h.*

For example, the property that expresses the persistency of the relation causing(PI1, P2, pos) is expressed in LEADSTO as follows:

causing(PI1, P2, pos) $\twoheadrightarrow_{0, 0, 1, 1}$ causing(PI1, P2, pos)

LEADSTO properties are specified in the LEADSTO Property Editor tool, implemented in SWI-Prolog/XPCE (see Figure 5). The Property Editor provides a user-friendly way of building and editing LEADSTO specifications. It was designed in particular for laymen. The tool has been used successfully by users with little computer experience. By means of graphical manipulation and filling in of forms a LEADSTO specification may be constructed. The end result is a saved LEADSTO specification file.

A LEADSTO specification in the Property Editor comprises different types of constructs among which:

- *Variables.* The language uses typed variables in various constructs. A variable is represented as <Var-Name>:<Sort>.
- *Sorts.* Sorts may be defined as a set of instances that may be specified: sortdef(<Sort-Name>, [<Term>,…]). There are also built-in sorts such as integer, real, and ranges of integers represented as for example between(8, 20).
- *Atoms.* Atoms may be terms built up from names with argument lists where each argument must be a term or a variable, for example: aggregation_of(pi1, pi2).

- *LEADSTO rules*. Represented as: leadsto([<Vars>,] <Antecedent-Formula>, <Consequent-Formula>, <Delay>, where <Delay> := efgh(<E-Range>,<F-Range>, <G-Range>,<H-Range>)); <Vars> := "[" <Variable>,... "]".
- *Time/Range*. Time and Range values occurring in LEADSTO rules and interval constructs may be any number or expression evaluating to a number.
- *Constants*. Constants may be defined using the following construct: constant(<Name>, <Value>). A constant(A, a(1)) entry in a specification will lead to A being substituted by a(1) everywhere in the specification.
- *Intervals*. Some atom values will be derived from LEADSTO rules. Others are not defined by rules but represent constant values of atoms over a certain time range. They are expressed as: interval([<Vars>,]<Range>,<LiteralConjunction>).

Since the LEADSTO specifications are executable, also the algorithms can be specified and executed using the LEADSTO, in particular the algorithm for performance evaluation specified in Section 6. The execution of a LEADSTO specification results into an execution trace, which can be stored and used further for analysis, considered in the following section.

In principle, no restriction on the complexity of a PI structure is enforced by the LEADSTO editor. However, the number of relations and the complexity of a PI structure have an impact on the automated verification, as it will be discussed in the following section.

8.2 Analysis of PI specifications using the TTL Checker tool

Execution traces generated by the LEADSTO environment, as well as empirical data collected e.g. by an enterprise management system can be analyzed by checking dynamic properties in the environment TTL Checker [7]. Such properties should be specified in TTL using the concepts and the relations of the performance-oriented view. In particular, the properties (constraints) that ensure the integrity and consistency of a PI structure defined in Sections 5 and 7 can be checked automatically on a PI specification represented by a LEADSTO execution trace using the TTL Checker environment.

The TTL checker environment has been implemented in SWI-Prolog/XPCE and consists of two closely integrated tools: the Property Editor and the Checker Tool. The Property Editor provides a user-friendly way of building and editing properties in TTL. By means of graphical manipulation and filling in forms a TTL specification can be constructed. User interaction with the tools involves three separate actions:

(1) Loading, editing, and saving a TTL specification in the Property Editor (see Figure 6).

(2) Loading and inspecting traces to be checked by activating the Trace Manager. Both, traces produced by the LEADSTO tool and empirical traces can be used for verification. Empirical traces provided to the TTL Checker may be obtained by formalizing empirical data from log-files produced by information systems or from results of experiments.

(3) Checking a property against a set of loaded traces by the Checker Tool. The property is compiled and checked, and the result is presented to the user.

Given a trace and a formalized property as an input, the automatic verification software generates a positive or negative result. A positive decision confirms 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.

The verification algorithm is a backtracking algorithm that systematically considers all possible instantiations of variables in the TTL formula under verification. Quantification over continuous or discrete time variables is replaced by quantification over a finite set of time intervals. In order to increase the efficiency of verification, the TTL formula that needs to be checked is compiled into a Prolog clause. Compilation is obtained by mapping conjunctions, disjunctions and negations of TTL formulae to their Prolog equivalents, and by transforming universal quantification into existential quantification. Thereafter, if this Prolog clause succeeds, the corresponding TTL formula holds with respect to all traces under consideration. The complexity of the algorithm has an upper bound in the order of the product of the sizes of the ranges of all quantified variables.

Thus, the verification algorithm has the polynomial complexity in the number of performance indicators and the relations between them in a PI structure, which is more efficient than model checking used for example in [26] that has exponential complexity.

The checking of the PI structure from the considered case study using the inference rules from Section 7 has shown that the PI structure is consistent.

9. Related Work on Organization Modeling

For creating models for organizations a number of approaches have been proposed in areas such as organization theory, computational enterprise modeling, and artificial intelligence. In the following paragraphs a brief literature review of such approaches is presented focused on a comparison to the general framework presented in this paper.

Organization models that are normally used in organization theory are represented by informal or semi-formal graphical descriptions that illustrate specific aspects of organizations [36, 42] (e.g., decision making, authority and power relations). Such models are usually quite intuitive and are normally based on extensive empirical studies of real organizations. However the disadvantages of such models are obvious: (1) lack of generality and relations between different specific types of models, and (2) graphically depicted data can not be effectively processed, combined and analyzed. A class of models built based on the system dynamics theory is an exception in organization theory devoid of both these disadvantages [23]. Organizational models specified in system dynamics are based on numerical variables and equations that describe how these variables change over time. Although such models can be computationally effective (i.e., used for simulations and computational analysis), nevertheless they still lack the ontological expressivity and the possibility for higher abstract representation that are needed to conceptualize wide range of relations and phenomena that exist in different types of organizations. A solution to this problem has been proposed in the area of computational enterprise modeling [e.g. 15, 24, 28, 20]. In this area a number of frameworks for enterprise engineering are introduced. Usually, these frameworks include enterprise reference architectures (i.e., sets of modeling concepts), modeling languages and methodologies for creating a detailed (semi-)formal representation of organizational structure, behavior, and the environment for different types of organizations. Within many of these frameworks different views on organizations are distinguished (e.g., information view, process view, resource view), for which specialized models are created. Such declarative enterprise models may be represented in information systems that automate different organization processes and allow interoperability between different parts of one enterprise and between enterprises. The framework discussed in this paper can be positioned in the area of computational enterprise modeling and has all these advantageous properties. As mentioned in Section 2, GERAM [28] provides a generalized template for the development of elaborated enterprise modeling frameworks, which was taken into account when the proposed framework has been developed.

Some frameworks [e.g. 20, 15] within the area of computational enterprise modeling propose automated techniques for analysis of specifications without properly defined semantics of the modelling language, however the results of such analysis are not completely reliable. Furthermore, the behaviour of such specifications may be unpredictable. For example, in the ARIS framework [20] the control flows are modelled using informal Event-driven Process Chains (EPCs), which limits the possibilities for analysis and its reliability. Similar observations can be made with respect to the framework described in [15].

The enterprise modeling approach presented in [46] considers four modeling views/aspects: structure, process, resources and goals which can roughly be mapped to the four views of the framework presented in this paper: the structure aspect corresponds to the organization-oriented view, the process aspect corresponds to the process-oriented view, the resource aspect corresponds to the agent-oriented view combined with the resource concept from the process-oriented view and the goals aspect corresponds to the performance-oriented view. The four aspects are modeled at three aggregation levels – strategy, organization and information system from which strategy and organization can be considered as aggregation levels as defined in the framework discussed here, while the framework discussed in this paper allows the specification of as many aggregation levels as necessary. The modeling languages are only presented in a semi-formal way. Furthermore, PIs and their relationships are not modeled in [46] and goals are only considered at a basic, to a certain extent implicit, level.

Some of the existing organization modeling frameworks make use of the i* approach [50] that focuses on the dependencies relationships between organizational actors. This approach recognizes both hard and soft goals and defines a (soft)goal dependency relationships between actors with respect to (soft)goals expressing that one actor depends on another to make a condition in the world come true. In comparison with the framework introduced in this paper goals are only informally specified; no format and unified representation is enforced. Positive and negative contribution to a different degree of tasks/goals to soft goals are modeled using contribution links.

Although some of these models are developed based on formal models of the concepts (i.e., formal ontologies) [24], they allow only limited possibilities for performing computational analysis (e.g., by simulations or by specific types of verification).

In order to enable more sophisticated types of analysis, techniques from mathematics and computer science may be used. In particular, operation research proposes mathematical methods for identifying best possible solutions to problems related to coordination and execution of the operations within an organization that improve or optimize the organizational performance [35].

Usually, operations research methods propose solutions to improve performance of an entire organization, rather than concentrating only on its specific elements.

At the same time other formal techniques exist, which are dedicated for analyzing particular aspects of an organization considered from a certain viewpoint (e.g., Petri-nets techniques used for modeling and analyzing business processes [19]).

Although such approaches can be useful and efficient, the scope of their application is limited to a particular view on an organization, based on a limited number of concepts. However, to perform a profound evaluation of the organizational performance and to enable analysis and prediction of organizational behavior under different influences, more sophisticated verification techniques should be used that employ concepts and relations between them across different views on organizations.

Furthermore, techniques from the area of artificial intelligence have been applied for modeling and analyzing multi-agent organizations [10, 18]. In such organizational models (software, hardware or human) agents are allocated to roles that stand in certain relations to each other and are described by sets of functionalities performed by an organization. Such models can be used for example for coordinating tasks execution in a multi-agent system [29], or for enforcing certain behaviors (e.g., normative systems) upon an agent system [48]. Although agent-based organizational models enable different types of formal analysis, most of these models are not able to capture the richness of the conceptual basis, social relationships and diversity of processes that exist in organizations of different types.

The goal of this work was to develop a framework for organizational modeling and analysis that possesses positive features of the modeling approaches described above, and provides means for elaborated manifold computational (agent-based) analysis of organizational models. In particular, the modeling languages defined in this framework are expressive enough to convey structures and processes of organizations of most types within different views on organizations. Furthermore, the framework enables computational analysis techniques for different aspects of organization structure and behavior that employ concepts and relations between these concepts across different views on organizations.

10. Related Literature on Performance Measurement

The area of performance measurement is an active field of research in management science attracting interest from both academic and practitioner circles. Researchers have been busy identifying and classifying important performance indicators for any company (e.g. [32]) and those relevant for different specific domains e.g. logistics, production, supply chains, etc. (e.g. [11, 14, 33, 47]). Results have also been reported on real-life case studies aimed at giving more insight on the relative importance and appropriateness of performance indicators in different situations. Originally only numerical, mostly financial, indicators were considered and emphasis was given to accounting-related methods such as activity-based costing (ABC) [17] which relates resource costs to activities, products and services so that a realistic view is given on the overall costs and profitability of the organization. Nowadays it is believed that non-financial and non-numerical indicators such as customer satisfaction, employee motivation, innovation, quality, market share can be very informative as well (e.g. [30]). The area has also developed in a different dimension from looking at single, independent performance indicators to considering systems of performance indicators where a notable example is the Balanced Scorecard of Kaplan and Norton [32]. Most of the modern performance management systems pay a special attention to stakeholders. In particular, Funk stresses in [25] the importance of creating a sustainable organization, which is ‘one whose characteristics and actions are designed to lead to a “desirable future state” for all stakeholders’. Furthermore, in [27] it is emphasized that ‘a performance measurement system should include systems for reviewing measures and objectives that make it possible both to adapt the system quickly to the changes in the internal and external contexts, and systematically to assess a company’s strategy in order to support continuous improvement’. The development of such systems is considered e.g. in [5, 6, 9]. As process management is becoming a part of the language and actions of many organizations many authors recognize the importance of process orientation of performance management systems [1]. The performance modeling framework proposed in this paper identifies explicit relations between organizational performance indicators, goals and processes.

An integrated performance measurement system that addresses many of the identified above aspects of performance measurement is considered in [5]. This system is based on four levels (Corporate, Business Units, Business Processes and Activities) and at each of these levels five key factors are considered (Stakeholders, Control Criteria, External Measures, Improvement Objectives and Internal Measures). This system allows defining the most appropriate types of performance measures, which are classified in internal, external, capability and learning measures. However, this approach does not consider relations between performance indicators (e.g., causality relations).

Another direction of research in the area of performance measurement is the research on aggregation of (often heterogeneous) performance indicators into a single measure of organizational performance [e.g. 16, 4]. The challenge here is how to balance measures with different measurement scales and importance which can be correlated or conflicting. This approach can easily be integrated in the general framework proposed in this paper as a more specific way of building aggregation expressions over performance indicators. Here the explicitly defined relationships between performance indicators in the framework can assist in building the expressions as for example with the 2-additive Choquet integral used in [4] which takes into account the interactions between pairs of indicators.

An alternative approach is taken in the area of multi-criteria decision analysis [49] where one of the goals is to rank alternatives (which could be measurements of organizational performance from a set of selected performance indicators) using outranking relations. Results coming from this line of research can be useful as background knowledge when building an organizational model.

Moreover, since performance measurement is a central issue for every organization, the organization's model should take it into account. In organization modeling, however, this is currently done implicitly at best. We are only aware of a few frameworks that satify this requirement. The integrated methodology GRAI/GIM [13,12] explicitly models performance indicators however only informally, in the context of decision making and without taking into account the relationships among the performance indicators. The relationships between performance indicators and goals (objectives) are also introduced informally without a precise definition. ARIS [20] models key performance indicators (KPIs) and allow for using the Balanced Scorecard approach for modeling cause-and-effect relationships and assign KPIs to the strategic objectives. However without formal foundations of the modeling approach the possibilities for analysis are limited.

Letier et al., on the other hand, define in [34] quality variables which can be related to the performance indicators defined in this approach in order to model partial degree of satisfaction of a goal. Based on them objective functions are defined which are used in the formulation of goals. A major difference is that in [34] probabilistic reasoning is used to determine the partial satisfaction of goals which is reflected in the definitions of objective functions and goals.

In [3] an approach is introduced to evaluate performance objectives of an enterprise based on measures of performance indicators defined using the fuzzy sets formalism. An objective is defined in this approach by a fuzzy set reflecting its vagueness and/or uncertainty. It is evaluated by comparing it to a measure that represents the observed value of a performance indicator. In contrast to this approach, the approach proposed in this paper considers goals (comparable to objectives in [3]) as required or desired states of an enterprise/individual characterized by a set of logical expressions that hold. Such expressions may be built based on not only logical operators (>, <, =), but also on functions over performance indicators (e.g., maximized, minimized) which is expressed through the notion of a goal pattern. Furthermore, expressions over multiple performance indicators can be used for formulating objectives, which is not considered in [3]. In addition, in our framework, relations between performance indicators are explicitly specified which is also not considered in [3].

It should be noted that sometimes measures such as customer satisfaction, profit, production costs, delivery time (typical performance indicators) are visible in other models as well – often in the definition of goals but they always remain implicit and the relationships between them are usually not discussed.

Performance evaluation as considered in this paper is related to the area of Business Activity Monitoring (BAM) which aims at real-time monitoring and analysis of business processes in order to improve speed and effectiveness of operations. [37] presents SENTINEL – a tool for supporting BAM based on semantic technologies, which includes an ontology for metrics (PIs in the terminology of this paper) and tools for computation and analysis of the metrics. The ontology does not consider relationships between PIs. Furthermore, goals are not modeled explicitly thus the strategic/operational targets and objectives of the company are not considered explicitly in the evaluation of performance which reduces the intuitiveness and clarity of the modeling and the interpretation of the results.

The work reported in this paper was inspired by the language defined in [41] however the language in [41] differs in a number of ways from the one introduced in this paper.

(1) In this paper the Temporal Trace Language (TTL) is used for formalizing the relationships which allows for temporal aspects of the relations to be represented as well, such as relationships that change their direction or strength with time – this is not possible using the language in [41].

(2) Furthermore, the language presented here was integrated in the general framework by introducing relationships connecting it to the rest of the performance-oriented view and the other views which enables performance evaluation and analysis. Using the whole framework allows for representing how performance indicators are related to processes in the workflow, the roles or agents to which the goals belong, as well as execution traces recording the actual executions of the model which can then be used to assess and analyze the performance.

(3) The language in [41] abstracts from the actual way in which the performance indicators are going to be measured and cannot be used for performance evaluation. In this paper the notion of a performance indicator was enriched by including a number of characteristics such as type, scale, unit of measurement, min/max value, time frame for evaluation, etc. which are used to guide the assessment of performance by defining the way a performance indicator is measured and evaluated. In addition, an algorithm is presented defining the procedure for evaluating the performance by determining the satisfaction of goals.

(4) In this work, the list of relationships was streamlined by focusing only on the most important ones – causality, correlation and aggregation – in order to not overcomplicate the model which is important in the frames of a general framework covering many other perspectives as well. The relationships were enriched by adding a degree of influence for causal relationships and inference rules to discover new relationships. Unlike [41], here the relationships are defined formally through axioms expressed in TTL.

(5) The expressivity of the notion of a goal pattern (which is roughly related to the notion of a quantifier in [41]) was extended with the types achieved/ceased and maintained/avoided which allow to reflect the difference between performance that should be achieved for a certain deadline and maintaining a specified level of performance for a certain time interval.

(6) The language in [41] is used for defining requirements which can have a different form from goals as considered in this paper. For example the qualified requirement in [41] can express a preference relation between qualified expressions on performance indicators.

11. Evaluation

In this Section the proposed approach is evaluated by comparing to other existing PI modeling approaches along three dimensions: modeling coverage; behavior and maintainability.

Modeling coverage

As discussed in Section 10, some modeling means proposed by our approach can be also found in other approaches. However, to our knowledge, none of the existing approaches provides such a wide spectrum of performance modeling means, as the proposed approach does. In the case study considered in this paper it has been illustrated how all the proposed modeling means can be applied to model subtleties of the organizational performance. The proposed modeling means can be divided roughly into three groups: means for modeling PI structures, means for modeling soft aspects of the organizational performance, and means for modeling relations with other organizational concepts. In the following each of these groups will be discussed in more detail.

The proposed approach allows modeling different types of influence relations between PIs, as well as composite structures of PIs. By modeling and analyzing PI structures one can gain a better understanding of direct and indirect interdependences that exist between PIs. This is particularly useful for the identification of sources of unsatisfactory organizational performance, as well as, determining ways for performance improvement. Commonly used approaches such as GRAI/GIM [12, 13] and Balanced Scorecard [32] do not consider relations between PIs.

Soft measures of performance are common for most existing organizations. Therefore, a special elaborated treatment of soft measures that differs from hard measures was described in this paper. Only a few existing approaches [3, 34] treat soft measures of performance in a formal way.

To understand sources and consequences of performance variability in an organization, in the proposed approach PIs are related to other organization modeling concepts: goals, processes, roles and agents. Some of the proposed relations were also investigated in other approaches: e.g., measures of performance and goals in [34]. However, the authors are not aware of any other existing formal framework that combines a performance-related view on an organization with other views commonly considered in Organization Theory (such as process-related, interaction-related, power-related). By modeling and analyzing relations between the views, one may identify (indirect) sources of performance inefficiencies and perform diverse automated experiments (e.g., simulation) aimed at performance improvement.

PI values often change over time. The modeling language TTL, used in the proposed approach, allows specifying temporal expressions over PIs (e.g., the rate of growth of the sales volume during some month is greater than some value; the number of clients is within some range during some time interval). Temporal properties over PIs can be checked automatically on actual data (recorded traces) of an organization using the TTL Checker software described in this paper. This enables different forms of temporal analysis of the organizational performance. Furthermore, TTL can be used to specify relations between PIs that change over time. The authors are not aware of any other existing PI modeling framework, in which the temporal dimension is treated formally with analysis possibilities.

Behavior

By defining relations between PIs and other organization modeling concepts explicitly, a method for the performance evaluation is proposed in this paper. In this method the organizational performance originates from organization processes and interactions between actors and is evaluated by determining the level of satisfaction of the key organizational objectives – goals. In other approaches that were considered [3, 34, 37] only parts of the performance evaluation cycle proposed in this paper and the corresponding relations between the organizational concepts were made explicit.

Maintainability

In this paper an approach is proposed to check the consistency of a PI structure. After adding or deleting PIs the structure may not be consistent any more. By running the semi-automated consistency check one can identify (possible) conflicts in the structure. We are not aware of any other approach for checking the consistency of PI structures.

12. Conclusion

This paper presents an approach for modeling performance indicators and the relationships between them which constitutes a part of an expressive general framework for organizational modeling and analysis. The proposed approach is part of the performance-oriented view of the framework which provides formal tools for analyzing organizational and individual performance and relating current performance to the organizational goals and their satisfaction as well as to tasks and processes of the organization. Due to its expressivity and formal basis the framework can be used in enterprise information systems. It also allows building structures that can be used for complex analysis both within the performance-oriented view and between the performance-oriented view and other views of the general framework. Some possibilities for analysis are mentioned here but will be elaborated and applied on larger case studies elsewhere. Other views and how they are related to each other will also be presented separately.

A number of directions for future work are possible. The application of the approach for the modeling and analysis of larger-scale real-world business scenarios will be necessary in order to further evaluate the framework and give more insight in its usefulness and the methodological issues of its application. Furthermore, more investigation is needed in the integration of the framework with the execution aspects of the supporting information systems in the context of performance evaluation.

References

[1] M.J. Benner and M.L Tushman. (2003), Exploitation, exploration, and process management: the productivity dilemma revisited. Academy of Management Review, 28, 238–256.

[2] P. Bernus, L. Nemes, and G. Schmidt, *Handbook on Enterprise Architecture*, Springer-Verlag, 2003.

[3] L. Berrah, G. Mauris, A. Haurat, and L. Foulloy, Global vision and performance indicators for an industrial improvement approach, Computers in Industry 43 (2000) 211-225.

[4] L. Berrah, G. Mauris, and J. Montmain, Monitoring the improvement of an overall industrial performance based on a Choquet integral aggregation, Omega 36(3) Special Issue on Multiple Criteria Decision Making for Engineering (2008) 340-351.

[5] U.S. Bititci, A.S. Carrie, and L. McDevitt, (1997). Integrated performance measurement systems: a development guide. International Journal of Operations and Production Management, 17, 522–534.

[6] U.S. Bititci, T. Turner, and C. Begemann, (2000). Dynamics of performance measurement systems. International Journal of Operations and Production Management, 20, 692–704.

[7] T. Bosse, C.M. Jonker, L. van der Meij, A. Sharpanskykh, and J. Treur (2006). Specification and Verification of Dynamics in Agent Models. International Journal of Cooperative Information Systems, vol. 18 (1), 167-193.

[8] T. Bosse, C. M. Jonker, L. van der Meij, and J. Treur (2007). A Language and Environment for Analysis of Dynamics by Simulation. International Journal of Artificial Intelligence Tools, vol. 16, 435-464, 2007

[9] M. Bourne, J. Mills, M. Wilcox, A.. Neely, and K. Platts, (2000). Designing, implementing and updating performance measurement systems. International Journal of Operation and production management, 20, 754–771.

[10] P. Bresciani, A. Perini, P. Giorgini, F. Giunchiglia, and J. Mylopoulos. Tropos: An Agent-oriented software development methodology. *Autonomous Agents and Multi-Agent Systems,* 8 (2004) 203-236

[11] P.C. Brewer and T.W. Speh. “Using the balanced scorecard to measure supply chain performance” *Journal of Business Logistics* 21(1), 2000, 75-93.

[12] D. Chen, B. Vallespir, and G. Doumeingts. GRAI integrated methodology and its mapping onto generic enterprise reference architecture and methodology, Computers in Industry 33 (1997) 387-394.

[13] G. Doumeingts, B. Vallespir, and D. Chen. “Decisional Modelling using the GRAI Grid”. In : Bernus, P., Mertins, K. and Schmidt, G. (Eds): *Handbook on Architectures of Information Systems*, Springer-Verlag (1998) 313-338

[14] F.T.S. Chan. “Performance measurement in a supply chain” *International Journal of Advanced Manufacturing Technology* 21(7), 2003, 534-548.

[15] CIMOSA – Open System Architecture for CIM; ESPRIT Consortium AMICE, Springer-Verlag, Berlin (1993)

[16] V. Clivillé, L. Berrah, and G. Mauris. “ Quantitative expression and aggregation of performance measurements based on the MACBETH multi-criteria method”, *International Journal of Production Economics, 105(1), 2007, 171-189.*

[17] R. Cooper, and R. S. Kaplan, Measure Cost Right: Make the Right Decisions, *Harvard Business Review*, September-October 1988, pp. 96-102.

[18] M. Dastani, J. Hulstijn, F. Dignum, and J.-J. Meyer. “Issues in Multiagent System Development”. In: *Proceedings of the Third International Joint Conference on Autonomous Agents and Multi Agent Systems AAMAS'04* (2004) 922-929

[19] R. David, H. Alla. “*Discrete, Continuous, and Hybrid Petri Nets*”. Springer-Verlag (2005)

[20] R. Davis, E. Brabänder. “*ARIS Design Platform*”. Springer, 2007.

[21] J. R. Evans. An exploratory study of performance measurement systems and relationships with performance results, Journal of operations Management, 22, 2004, pp.219-232.

[22] ENV 12204: Advanced Manufacturing Technology - Systems Architecture - Constructs for Enterprise Modelling, CEN TC 310/WG1, 1996

[23] J.W. Forrester. “*Industrial dynamics*”, Waltham, MA: Pegasus Communications, 1961.

[24] M.S. Fox. “The TOVE project: towards a common-sense model of the enterprise”. In Petrie, C.J., Jr.(Ed): *Proceedings of ICIEMT’92*, MIT Press (1992) 310-319

[25] K. Funk, Sustainability and performance. MIT Sloan Management Review, 44, 65–71 (2003)

[26] A. Fuxman, J. Mylopoulos, M. Pistore, P. Traverso, Model Checking Early Requirements Specifications in Tropos. In Proceedings of the Fifth IEEE International Symposium on Requirements Engineering (RE'01), 2001, pp. 174-181.

[27] P. Garengo, S. Biazzo and U. Bititci, Performance measurement systems in SMEs: A review for a research agenda, International Journal of Management Reviews 7 (2005), pp. 25–48.

[28] GERAM: The Generalized Enterprise Reference Architecture and Methodology. IFIP-IFAC Task Force on Architectures for Enterprise Integration, In : Bernus, P., Nemes, L. and Schmidt, G. (Eds): *Handbook on Enterprise Architectures*, Springer-Verlag (2003) 21-63

[29] B. Horling, and V. Lesser. “A Survey of Multi-Agent Organizational Paradigms”. *The Knowledge Engineering Review*, Vol 19(4), Cambridge University Press, 2005, pp. 281-316.

[30] C.D. Ittner and D.F. Larcker. “Coming Up Short on Nonfinancial Performance Measurement” *Harvard Business Review*, 81(11), 2003, 88-96.

[31] C.M. Jonker, A. Sharpanskykh, J. Treur, P. Yolum: A Framework for Formal Modeling and Analysis of Organizations, Applied Intelligence, 27(1), 49-66 (2007)

[32] R.S. Kaplan, and D.P. Norton. “The balanced scorecard – measures that drive performance”, *Harvard Business Review*, January-February 1992, pp. 71-79.

[33] E. Krauth, H. Moonen, V. Popova, and M. Schut. “Performance Measurement and Control in Logistics Service Providing”. In: C.-S. Chen, J. Filipe, I. Seruca and J. Cordeiro, editors, *Proceedings of Seventh International Conference on Enterprise Information Systems, ICEIS 2005*, pp. 239-247.

[34] E. Letier, and A. van Lamsweerde. “Reasoning about partial goal satisfaction for requirements and design engineering”, *Proceedings of 12th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, 2004, pp. 53 – 62.

[35] W. H. Marlow. “*Mathematics for Operations Research*”. New York: Dover (1993)

[36] H. Mintzberg: The Structuring of Organizations. Prentice Hall, Englewood Cliffs, N.J. (1979)

[37] C. Pedrinaci, D. Lambert, B. Wetzstein, T. van Lessen, L. Cekov, M. Dimitrov. SENTINEL: A Semantic business process monitoring tool. 1st International Workshop on Ontology-supported Business Intelligence

(OBI2008), in conjunction with 7th International Semantic Web Conference (ISWC2008), Karlsruhe, Germany, October 27, 2008.

[38] V. Popova and A. Sharpanskykh. “Formal Analysis Of Executions Of Organizational Scenarios Based On Process-Oriented Models”. In*: Proceedings of 21st European Conference on Modelling and Simulation ECMS 2007,* SCS Press, pp. 36-44.

[39] V. Popova and A. Sharpanskykh. Formal goal-based modeling of organizations. In: *Proceedings of the 6th International Workshop on Modelling, Simulation, Verification and Validation of Enterprise Information Systems (MSVVEIS’08),* INSTICC Press, 2008.

[40] V. Popova, A. Sharpanskykh, Performance-Oriented Organization Modelling. Technical Report 062109AI, Vrije Universiteit Amsterdam, http://hdl.handle.net/1871/10213

[41] Popova, V., Treur J. A specification language for organizational performance indicators", Applied Intelligence Journal, Volume 27, Number 3, pp.291-301 / December, 2007

[42] W.R. Scott, *Institutions and organizations*. 2nd edn, SAGE Publications, Thousand Oaks London New Delhi, 2001.

[43] A. Sharpanskykh, Authority and its implementation in enterprise information systems. In: Sadiq, S., Reichert, M., Schulz, K., Trienekens, J., Moller, C., and Kusters, J. (eds.), Proceeding of the 1st International Workshop on Management of Enterprise Information Systems, MEIS 2007, INSTICC Press, 33-43 (2007)

[44] A. Sharpanskykh, Modeling of agents in organizational context. In: H-D. Burkhard, G. Lindeman, L. Varga, R. Verbrugge (eds.), Proceedings of the 5th International Central and Eastern European Conference on Multi-Agent Systems, LNAI 4696, Springer Verlag, pp. 193-204.

[45] A. Sharpanskykh, and J.Treur, Verifying interlevel relations within multi-agent systems, in *Proceedings of the 17th European Conference on Artificial Intelligence, ECAI'06*, IOS Press, 2006, pp. 290-294.

[46] Frank, U. 2002. Multi-perspective Enterprise Modeling (MEMO) – Conceptual Framework and Modeling Languages. In Proceedings of the 35th Annual Hawaii international Conference on System Sciences (Hicss'02)-Volume 3 - (January 07 - 10, 2002). HICSS. IEEE Computer Society, Washington, DC, 72.

[47] G. Vaidyanathan. A Framework for Evaluating Third-Party Logistics *Communications of the ACM*, January 2005 48, 1: 89-94.

[48] J. Vázquez-Salceda, H.M. Aldewereld and F.P.M. Dignum. “Norms in multiagent systems: From theory to practice”. *International Journal of Computer Systems Science & Engineering*, 20(4) (2005) 225-236

[49] Ph. Vincke, *Multicriteria Decision Aid*, Wiley, New York.

[50] E. Yu, Towards Modelling and Reasoning Support for Early-Phase Requirements Engineering. 3rd IEEE Int. Symposium on Requirements Engineering (1997) 226-235

Appendix A. A specification of the performance indicators from the case study

Name: **PI1**

Definition: the level of correctness of plans with respect to the contracts of the employees, the laws, the general policy of the company and division

type: discrete

time frame: month

scale: very_low-low-medium-high-very_high

source: mission statement, job descriptions

owner: forward and daily planning departments

threshold: 2 units

hardness: soft

Name: **PI2**

Definition: the level of knowledge of employees involved in (forward) planning about the current contracts of the employees, the laws, the general policy of the company and division

type: discrete

time frame: month

scale: very_low-low-medium-high-very_high

source: mission statement, job descriptions

owner: forward and daily planning departments

threshold: 2 units

hardness: soft

Name: **PI3**

Definition: the level of up-to-dateness of the software system used in (forward and daily) planning with respect to the contracts of the employees, the laws, the general policy of the company and division

type: discrete

time frame: month

scale: very_low-low-medium-high-very_high

source: mission statement, job descriptions

owner: forward and daily planning departments

threshold: 2 units

hardness: soft

Name: **PI4**

Definition: effectiveness of allocation of security officers

type: discrete

time frame: month

scale: very_low-low-medium-high-very_high

source: mission statement, job descriptions

owner: forward and daily planning departments, unit manager, security officers

threshold: 2 units

hardness: soft

Name: **PI5**

Definition: average correctness of produced plans

type: discrete

time frame: month

scale: very_low-low-medium-high-very_high

source: mission statement, job descriptions

owner: forward and daily planning departments

threshold: 2 units

hardness: soft

Name: **PI6**

Definition: level of correctness of every produced plan

type: discrete

time frame: month

scale: very_low-low-medium-high-very_high

source: job descriptions

owner: forward and daily planning departments

threshold: 2 units

hardness: soft

Name: **PI7**

Definition: level of correctness of every produced forward plan.

type: discrete

time frame: month

scale: very_low-low-medium-high-very_high

source: job descriptions

owner: forward planning department

threshold: 2 units

hardness: soft

Name: **PI8**

Definition: level of correctness of every produced daily plan.

type: discrete

time frame: day

scale: very_low-low-medium-high-very_high

source: job descriptions

owner: daily planning departments

threshold: 2 units

hardness: soft

Name: **PI9**

Definition: level of correctness of every produced short-term plan.

type: discrete

time frame: week

scale: very_low-low-medium-high-very_high

source: job descriptions

owner: daily planning departments

threshold: 2 units

hardness: soft

Name: **PI10**

Definition: level of correctness of administrative processing of all planning data in the system.

type: discrete

time frame: month

scale: very_low-low-medium-high-very_high

source: job descriptions

owner: forward and daily planning departments

threshold: 2 units

hardness: soft

Name: **PI11**

Definition: level of correctness of administrative processing of forward planning data in the system.

type: discrete

time frame: month

scale: very_low-low-medium-high-very_high

source: job descriptions

owner: forward planning department

threshold: 2 units

hardness: soft

Name: **PI12**

Definition: level of correctness of administrative processing of short-term and daily planning data in the system.

type: discrete

time frame: month

scale: very_low-low-medium-high-very_high

source: mission statement, job descriptions owner: daily planning departments

threshold: 2 units

hardness: soft

Name: **PI13**

Definition: average level of deviation from daily plans in their application.

type: discrete

time frame: month

scale: very_low-low-medium-high-very_high

source: job descriptions

owner: daily planning departments, unit manager, security officers

threshold: 2 units

hardness: soft

Name: **PI14**

Definition: level of deviation from the produced daily plan in its application

type: discrete

time frame: day

scale: very_low-low-medium-high-very_high

source: job descriptions

owner: daily planning departments, unit manager, security officers

threshold: 2 units

hardness: soft

Name: **PI15**

Definition: the number of concerned security officers not informed on time about the produced daily plan

type: discrete

time frame: day

scale: INTEGER

min value: 0

max value: max_officers

unit: employees

source: job descriptions

owner: daily planning departments

threshold: 10

hardness: hard

Name: **PI16**

Definition: promptness of data change forms delivery by security officers

type: discrete

time frame: day

scale: very_low-low-medium-high-very_high

source: job descriptions

owner: security officers

threshold: 2 units

hardness: soft

Name: **PI17**

Definition: promptness of data change forms delivery to the planners

type: discrete

time frame: day

scale: very_low-low-medium-high-very_high

source: job descriptions

owner: unit manager

threshold: 2 units

hardness: soft

Name: **PI18**

Definition: level of correctness of data change forms delivered to the planners

type: discrete

time frame: day

scale: very_low-low-medium-high-very_high

source: job descriptions

owner: unit manager

threshold: 2 units

hardness: soft

Name: **PI19**

Definition: efficiency of planning and allocation of security officers

type: discrete

time frame: month

scale: very_low-low-medium-high-very_high

source: mission statement, job descriptions

owner: forward and daily planning departments

threshold: 2 units

hardness: soft

Name: **PI20**

Definition: efficiency of the planning process

type: discrete

time frame: month

scale: very_low-low-medium-high-very_high

source: mission statement, job descriptions

owner: forward and daily planning departments

threshold: 2 units

hardness: soft

Name: **PI21**

Definition: the number of times the planning activities (create and update plans) exceed the allowed durations

type: continuous

time frame: month

scale: REAL

min value: 0

max value: max_time

unit: hour

source: job descriptions

owner: daily planning departments

threshold: 24 hours

hardness: hard

Name: **PI22**

Definition: the time to update a short-term plan given operational data

type: continuous

time frame: month

scale: REAL

min value: 0

max value: max_time_UST

unit: hour

source: job descriptions

owner: daily planning departments

threshold: 24 hours

hardness: hard

Name: **PI23**

Definition: the time to create a daily plan given operational data

type: continuous

time frame: day

scale: REAL

min value: 0

max value: max_time_D

unit: hour

source: job descriptions

owner: daily planning departments

threshold: 12 units

hardness: hard

Name: **PI24**

Definition: level of promptness of communication of forward, short-term and daily planning data to all concerned employees.

type: discrete

time frame: month

scale: very_low-low-medium-high-very_high

source: job descriptions

owner: forward and daily planning departments

threshold: 2 units

hardness: soft

Name: **PI25**

Definition: level of promptness of communication of every produced forward plan to all concerned employees.

type: discrete

time frame: month

scale: very_low-low-medium-high-very_high

source: job descriptions

owner: forward planning department

threshold: 2 units

hardness: soft

Name: **PI26**

Definition: level of promptness of communication of every produced short-term plan to all concerned employees.

type: discrete

time frame: month

scale: very_low-low-medium-high-very_high

source: job descriptions

owner: daily planning departments

threshold: 2 units

hardness: soft

Name: **PI27**

Definition: the time to create a short-term plan after all operational data is received

type: continuous

time frame: month

scale: REAL

min value: 0

max value: max_time_CST

unit: hour

source: job descriptions

owner: daily planning departments

threshold: 24 hours

hardness: hard

Name: **PI28**

Definition: the time to create a forward plan after all operational data is received

type: continuous

time frame: month

scale: REAL

min value: 0

max value: max_time_CFP

unit:

source: job descriptions

owner: forward planning department

threshold: 48 units

hardness: hard

Name: **PI29**

Definition: efficiency of allocation of security officers

type: discrete

time frame: month

scale: very_low-low-medium-high-very_high

source: mission statement, job descriptions

owner: forward and daily planning departments

threshold: 2 units

hardness: soft

Name: **PI30**

Definition: average level of optimality of forward, short-term and daily planning for efficient allocation of security officers

type: discrete

time frame: month

scale: very_low-low-medium-high-very_high

source: job descriptions

owner: forward and daily planning departments

threshold: 2 units

hardness: soft

Name: **PI31**

Definition: average level of optimality of every forward plan for efficient allocation of security officers

type: discrete

time frame: month

scale: very_low-low-medium-high-very_high

source: job descriptions

owner: forward planning department

threshold: 2 units

hardness: soft

Name: **PI32**

Definition: average level of optimality of every short-term plan for efficient allocation of security officers

type: discrete

time frame: month

scale: very_low-low-medium-high-very_high

source: job descriptions

owner: daily planning departments

threshold: 2 units

hardness: soft

Name: **PI33**

Definition: level of optimality of every daily plan for efficient allocation of security officers

type: discrete

time frame: month

scale: very_low-low-medium-high-very_high

source: job descriptions

owner: daily planning departments

threshold: 2 units

hardness: soft

Figure 1: The process of extracting performance indicators

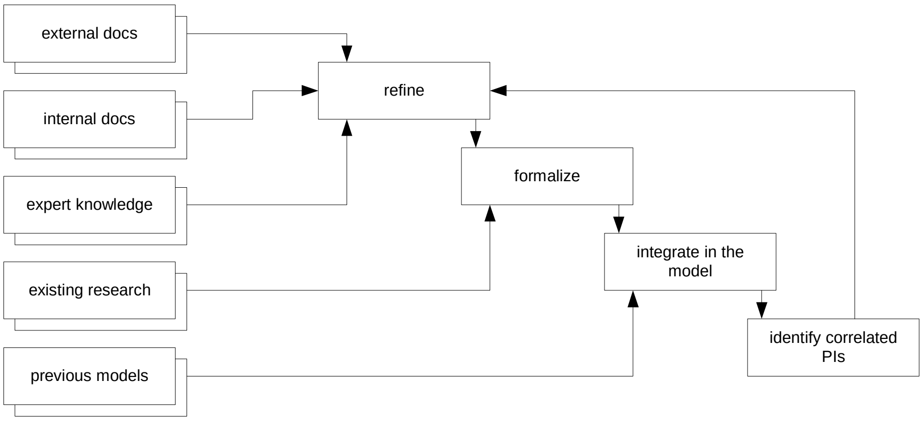


Figure 2. A meta-model for the *performance-oriented view*.

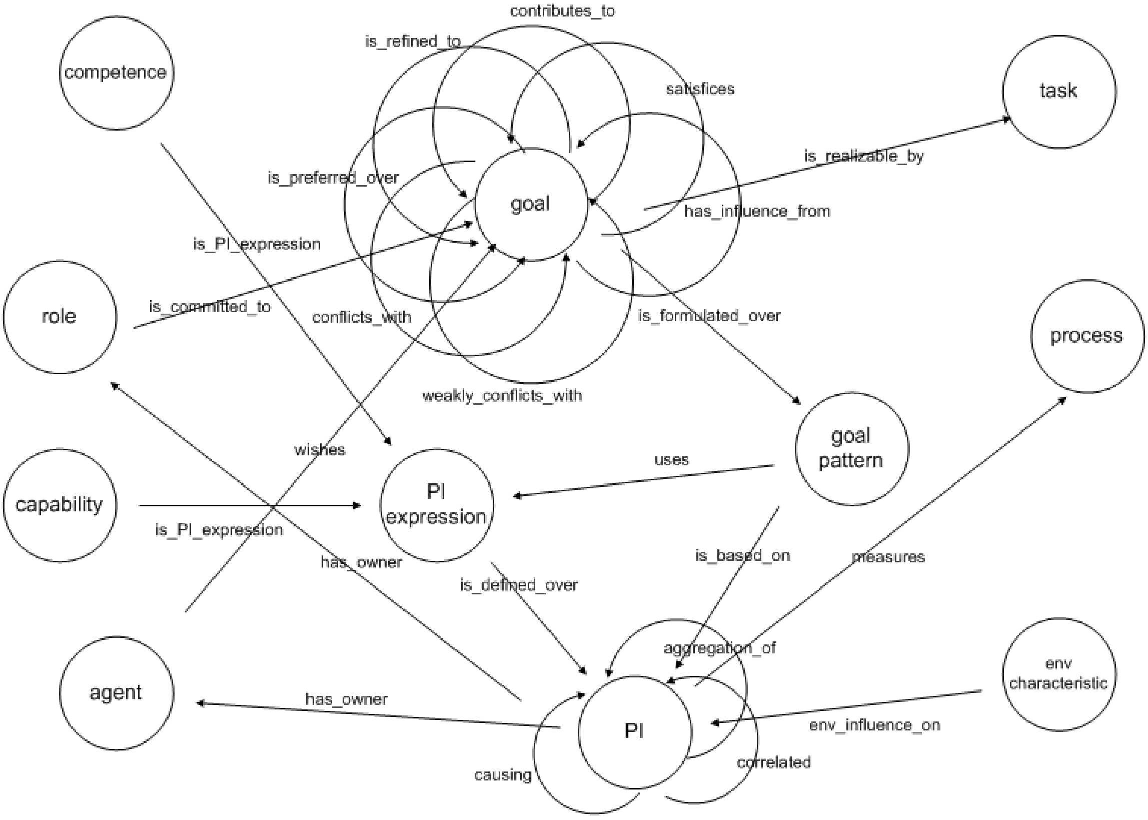


Figure 3: The relationships between the performance indicators identified for the case study

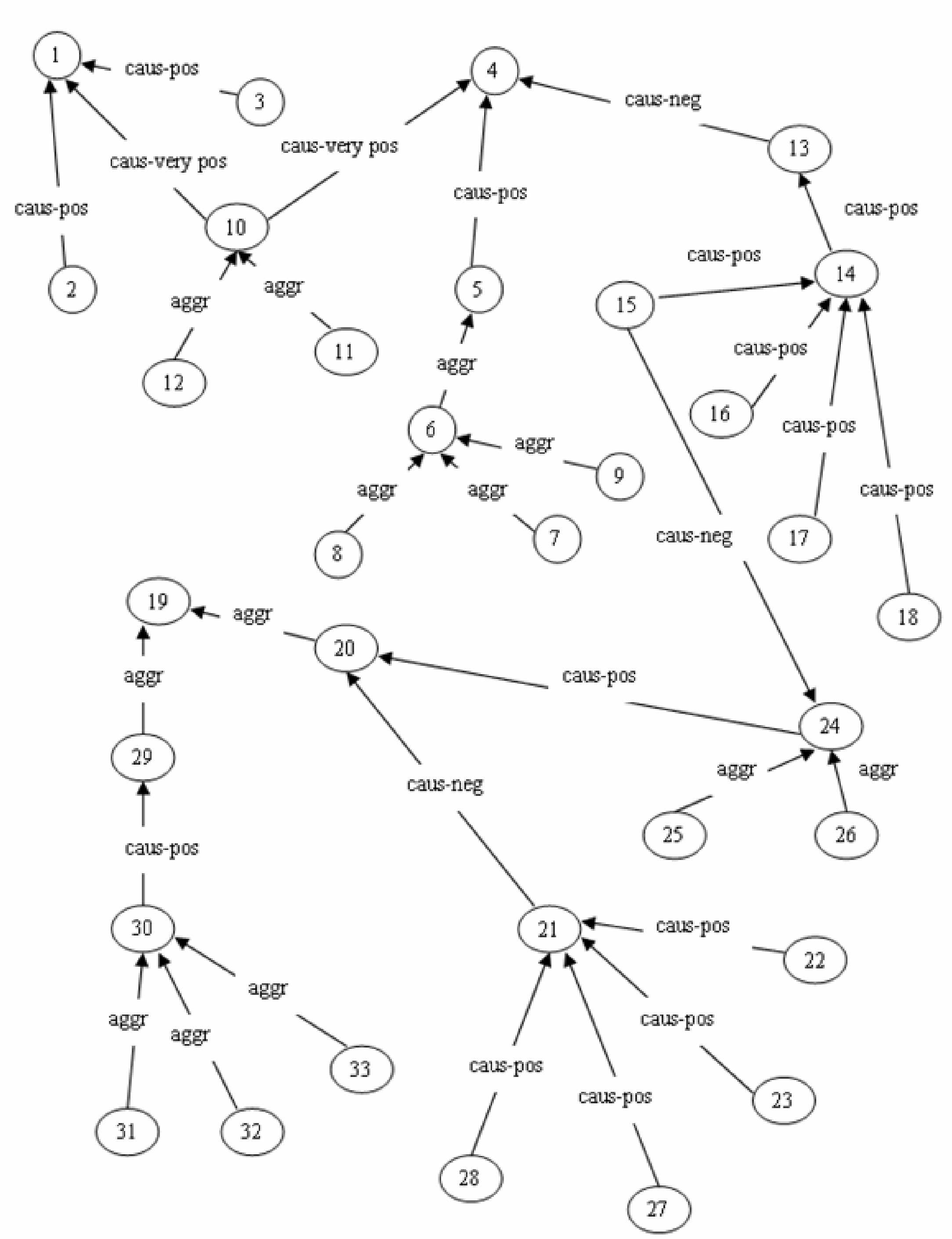


Figure 4: Timing relationships in LEADSTO

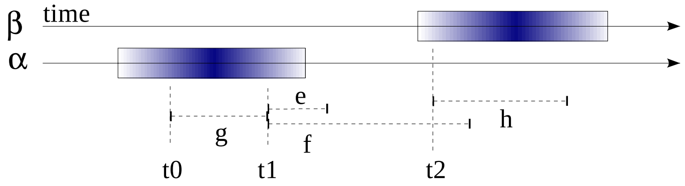
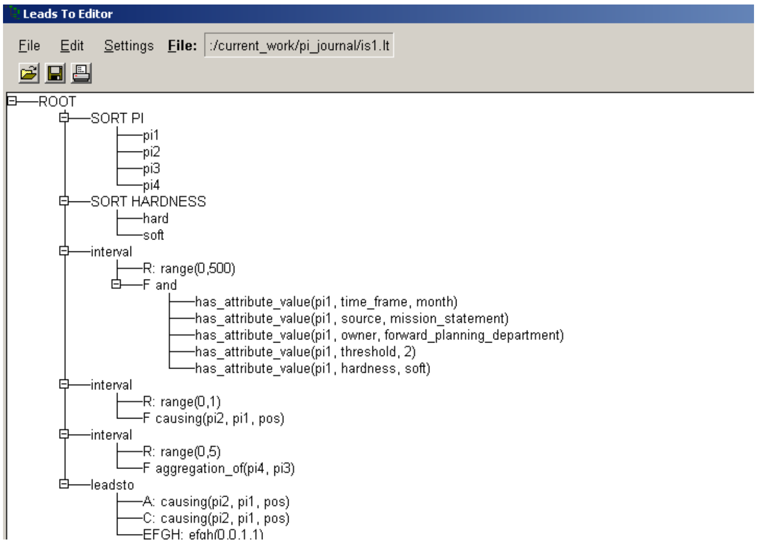
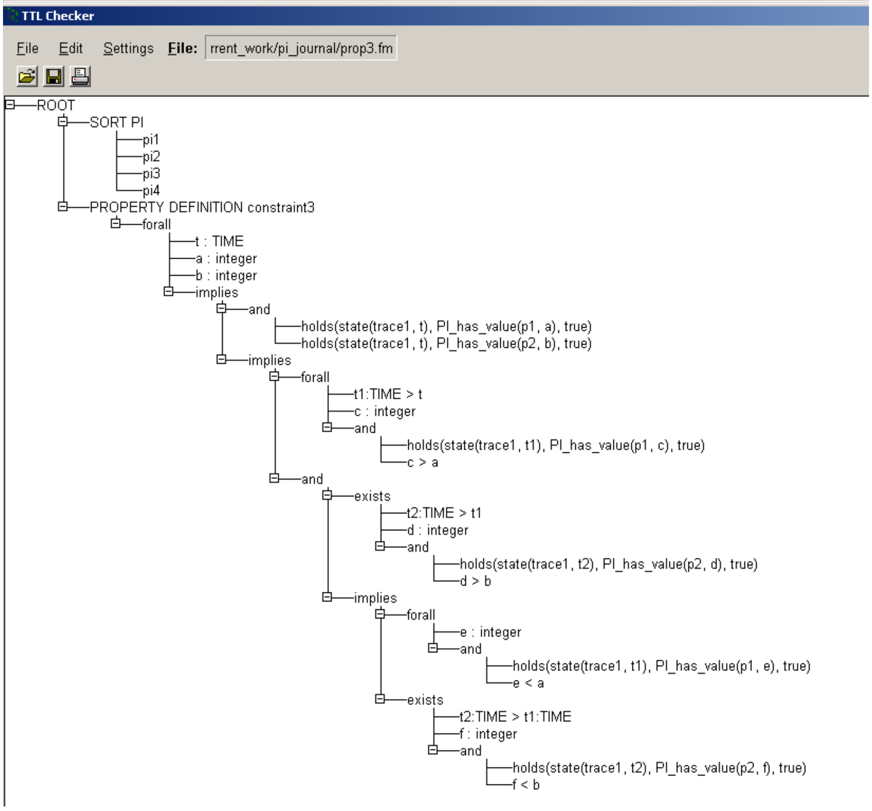


Figure 5: The LEADSTO Property Editor



```
ROOT
    SORT PI
        pi1
        pi2
        pi3
        pi4
    SORT HARDNESS
        hard
        soft
    interval
        R: range(0,500)
        F and
            has_attribute_value(pi1, time_frame, month)
            has_attribute_value(pi1, source, mission_statement)
            has_attribute_value(pi1, owner, forward_planning_department)
            has_attribute_value(pi1, threshold, 2)
            has_attribute_value(pi1, hardness, soft)
    interval
        R: range(0,1)
        F causing(pi2, pi1, pos)
    interval
        R: range(0,5)
        F aggregation_of(pi4, pi3)
    leadsto
        A: causing(pi2, pi1, pos)
        C: causing(pi2, pi1, pos)
        EFGH: efgh(0,0,1,1)
```

Figure 6: The TTL Checking Environment



File: rrent_work/pi_journal/prop3.fm

```
ROOT
  SORT PI
    pi1
    pi2
    pi3
    pi4
  PROPERTY DEFINITION constraint3
    forall
      t : TIME
      a : integer
      b : integer
      implies
        and
          holds(state(trace1, t), PI_has_value(p1, a), true)
          holds(state(trace1, t), PI_has_value(p2, b), true)
        implies
          forall
            t1:TIME > t
            c : integer
            and
              holds(state(trace1, t1), PI_has_value(p1, c), true)
              c > a
          and
            exists
              t2:TIME > t1
              d : integer
              and
                holds(state(trace1, t2), PI_has_value(p2, d), true)
                d > b
            implies
              forall
                e : integer
                and
                  holds(state(trace1, t1), PI_has_value(p1, e), true)
                  e < a
              exists
                t2:TIME > t1:TIME
                f : integer
                and
                  holds(state(trace1, t2), PI_has_value(p2, f), true)
                  f < b
```

Table 1. Inference Rules for Causal Relationships

s2 \ s1	**Very neg**	**Neg**	**Pos**	**Very pos**
Very neg	Very pos	Very pos	Very neg	Very neg
Neg	Very pos	Pos	Neg	Very neg
Pos	Very neg	Neg	Pos	Very pos
Very pos	Very neg	Very neg	Very pos	Very pos