BULLETIN OF THE POLISH ACADEMY OF SCIENCES
TECHNICAL SCIENCES, Vol. 67, No. 1, 2019
DOI: 10.24425/bpas.2019.127333

Data processing in self-controlling enterprise processes

M. ZABOROWSKI*

Department of Management Engineering, the University of Dąbrowa Górnicza, 1c Cieplaka St., 41-300 Dąbrowa Górnicza, Poland

Abstract. This study presents cause-effect dependencies between inputs and outputs of business transitions that are software objects designed for processing information-decision state variables in integrated enterprise process control (EntPC) systems. Business transitions are elementary components of controlling units in enterprise processes that have been defined as self-controlling, generalized business processes, which may serve not only as business processes but also as business systems or their roles. Business events, which have zero durations by definition, are interpreted as executions of business actions that are main operations of business transitions. Any ordered set of business actions, performed in the controlling unit of a given enterprise process and attributed to the same discrete-time instant, is referred to as 'the information-decision process'. The i-d processes may be substituted by managerial business processes, performed on the lower organizational level, where durations of activity executions are greater than zero, but discrete-time periods are considerably shorter. In such a case, procedures of business actions are performed by corresponding activities of managerial processes, but on the level of business transitions the durations of their executions are imperceptible, and many different business events may occur at the same discrete-time instant. It has been demonstrated in the paper how to control business actions to ensure that a given i-d state variable may not change more than once at a given instant. Furthermore, the rules of designing the i-d process structures, which prevent random changes of transitory states, have been presented.

Key words: enterprise process control, complex control systems, industry 4.0, enterprise process state, enterprise integration.

1. Introduction. Towards the theory of enterprise process control

The primary reason for beginning research work on creating a theory of enterprise process control (EntPC) was a need for such a reference model for integrated management and process control (IMPC) systems in which it would be possible to compare alternative decision-making methods applied to the same enterprise model [1]. So, the first area of EntPC theory is a formal description of the framework EntPC system in which one can replace individual decision algorithms and other information processing procedures without changing its structure. Concrete IMPC systems, whose structures are conformable with the EntPC reference model, have been named 'enterprise process control systems' [2] (EntPC systems). The thesis on universality of EntPC theory [2] claims that every IMPC system, irrespective of the industry and the size of the enterprise in which it is implemented, may be replaced, retaining all its functions and data, with a corresponding EntPC system. To clarify, this concerns all functions of ERP, MES, SCADA and PLC systems that belong to the IMPC systems described by the ISA-95 standard [3].

Control is generally defined as a goal-oriented action of an object, referred to as a controlling unit (e.g. a controller or a controlling system), upon another object, referred to as a control plant [4]. Management, perceived as influencing somebody to do something, is also a special case of control. According to the first sentence of the celebrated handbook of control theory, control plants are processes [5]. In EntPC theory, they are infrastructural processes, which are control plants of base direct control systems, as well as sets of business activities, which are subordinate business processes [2].

As per the APICS dictionary [6], a 'business process' is a set of logically related tasks or activities performed to achieve a defined business outcome. And as per [7], 'a business process is one focused upon the production of particular products. These may be … physical products. … The "product" can also be a service'. Hence, business processes are divided into manufacturing processes and service processes. Workflow processes, whose products are documents [8], are counted among service processes [9].

In EntPC theory, a self-controlling business process has been defined as a system of control for a finite, partially ordered set of enterprise activities, which transform their input business products into output products to fulfil the requirements of other business processes, belonging to a given enterprise or to its environment [2, 10]. Business products may be material resources, services or documents. Business activities are stages of business processes. Conversely, every business process watched from the outside is a business activity of the higher level.

Business systems, i.e. systems designed for performing business processes, are defined as control systems, whose control plants are sets of all business processes performed by them. A business system's role is defined analogously as a control system of jointly managed business processes, distinguished in the business system according to the competence and authority or the resources that are required.

*e-mail: zaborowski.miroslaw@gmail.com

Manuscript submitted 2017-12-17, revised 2018-03-13, initially accepted for publication 2018-03-19, published in February 2019.

In the case of an EntPC system, control is decentralized in the hierarchical organizational structure, where control plants may be subordinate control systems [11], and in the multistage structure of types of transactions [12] between delivery and receiving processes (Fig. 1). The component control systems of an entire EntPC system are self-controlling enterprise processes (Fig. 2).

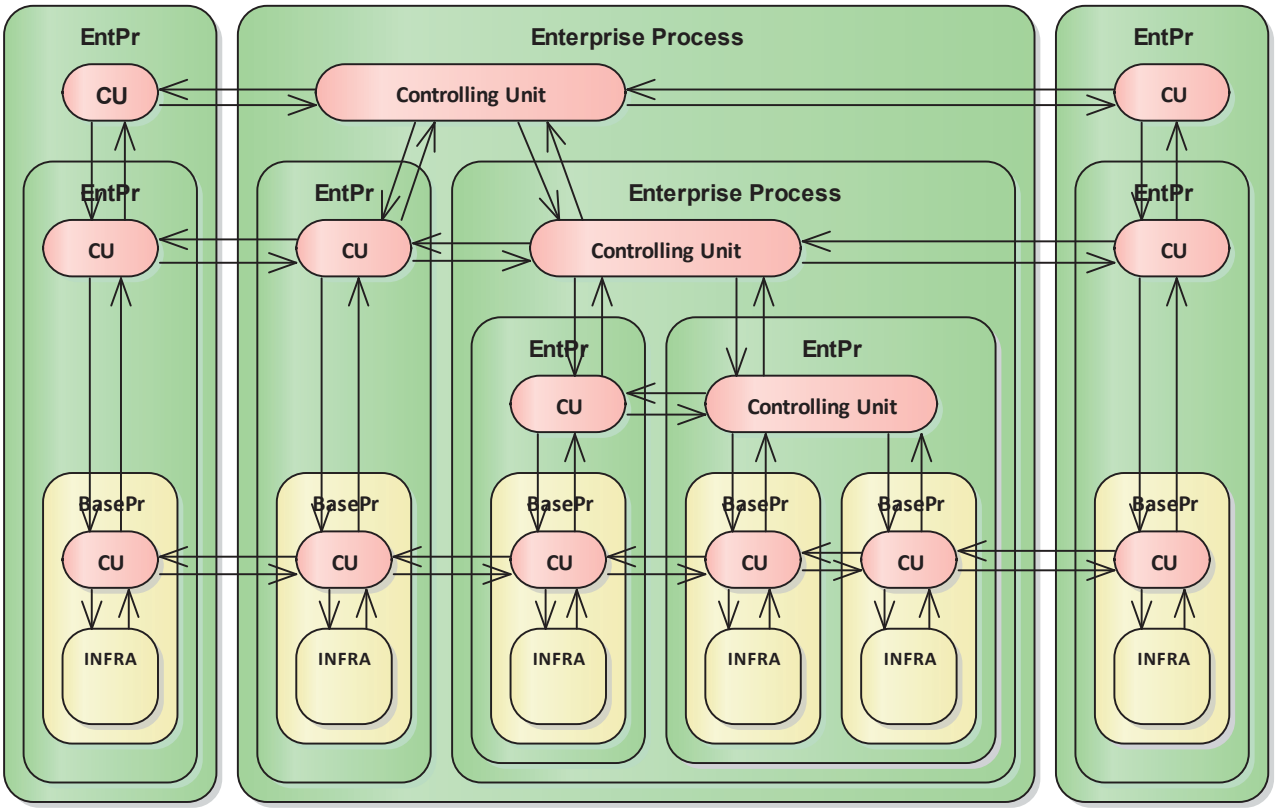

Fig. 1. Sketch of hierarchical and transactional couplings between control systems of enterprise business and base processes

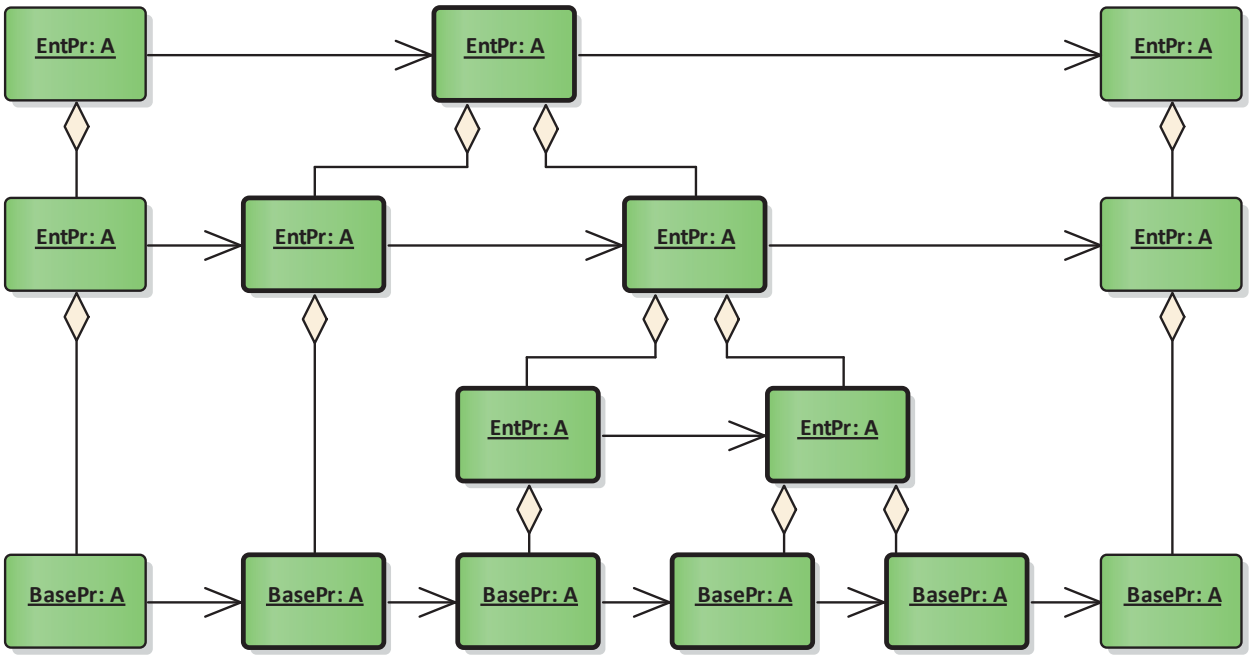

Fig. 2. Sketch of aggregation and order relationships between activities of enterprise processes in the form of an UML object diagram

Enterprise processes are base processes, i.e. control systems whose controlled plants are infrastructural processes, or generalized business processes, i.e. business processes themselves, as well as business systems and their roles. Any enterprise process is a system of control for a finite, partially ordered set of enterprise activities. Enterprise activities, $a \in A$, are stages of enterprise processes. Conversely, every enterprise process watched from the outside is an enterprise activity, $p \in P \subset A$, of the higher level. An enterprise itself is a process at the top of hierarchy of its EntPC system (Fig. 2). Self-controlling of enterprise processes is an essential distinguishing feature of the EntPC systems because it enables effective influencing of the course of processes, whereas business process management systems, conformable with popular YAWL and BPMN standards [8, 13], can only control launching executions of business activities and monitor their endings.

Enterprise activities are generalized business activities that may be business activities, $a \in Aa$, and business units, $a \in U$, as well as their roles, $a \in G$:

$$a \in A = Aa \cup U \cup G.$$

In EntPC systems, the identification numbers of software objects are equated with these objects [14]. Business units, i.e. business systems watched from the outside, are equated with collective business activities, $a \in U$, which are sets of all business activities performed by these units, because they have the same identifiers. For the same reason, business roles, i.e. business systems roles watched from the outside, are equated with corresponding sets of jointly managed business activities.

It has been demonstrated that all enterprise objects, including structural objects (business units, enterprise activities, business accounts, business products, business tasks, products of business tasks and the like) and their attributes (business variables, functional variables, information-decision state variables and their records), belong to the enterprise structure tree of definite composition relationships [2, 15]. The root of this structure tree is the enterprise as a whole. Business agents, $k \in Kb$, that are designed for data processing in EntPC systems also belong to this structure tree as components of enterprise activities.

Any EntPC system may be regarded as a complex control system with one huge controlling system, which is a network of controlling units of self-controlling enterprise processes, and one control plant in the form of the set of infrastructural processes (Fig. 1). Such an approach may prove particularly useful for the industry 4.0 enterprises [16–19], because their management systems should react in real time to the enterprise state changes [17, 20]. On the other hand, however, real-time control systems are the subject of control theory. The general mathematical model of EntPC controlling systems in the form acceptable in control theory may facilitate transferring the results of this theory to the systems of enterprise management, e.g. in order to analyze stability and controllability of an enterprise, regarded as a complex switched system [21], or to assess management quality using criteria and methods applied to control systems.

The objective of the paper is to demonstrate that the set of function dependencies between input and output variables of procedures performed in the controlling system of a given EntPC system, as well as in the controlling units of its enterprise processes, may be presented in the form of the general mathematical model of the controller in a discrete control system (equations (2.15) (2.16) in [4] with changed notation):

$$\boldsymbol{x}_n = \boldsymbol{f}(\boldsymbol{x}_{n-1}, u_n) \tag{1}$$

$$v_n = \eta(\boldsymbol{x}_n), \tag{2}$$

where u_n, v_n are, correspondingly, vectors of input and output variables of the controller algorithm in the period n and x_n is its state vector at the end of this period. This thesis is not obvious because model (1) (2) is valid for the one, sole algorithm, whereas in practice there are many different procedures whose executions are assigned to the same discrete time instant.

The paper is organized as follows. In the introduction, the main concepts and the scope of practical applications of the author's EntPC theory are presented. The functional structure of EntPC systems, including the internal structure of controlling units of enterprise processes, is discussed in chapter 2. In chapter 3, the information-decision state of EntPC systems is defined. Moreover, relationships between i-d state variables and structural objects of the EntPC conceptual model are outlined. Chapter 4 describes the structure of the cause-effect dependencies between inputs and outputs of business transitions that are software objects designed for processing i-d state variables. The requirements concerning duration and sequence of business transition executions are discussed in chapter 5. Furthermore, it is shown how business activities of managerial processes can substitute procedural business transitions in the case of extensively long execution durations. Finally, chapter 6 concludes the paper.

2. Structure of self-controlling enterprise processes

2.1. Self-controlling business processes and base processes. The classical structure of a simple direct control system [22] is a feedback loop consisting of a control plant, a measurement device, a controller and an actuating device. In a simplified form, measurement and actuating devices are hidden in the control plant [5]. However, the structure of a feedback control system may also be presented in another way (Fig. 3), facilitating its use for describing business process control systems. In this structure, only the actuating device is included in the control plant, whereas the measurement device is presented as forming part of the controlling unit. Thus, control encompasses both:

- acquisition of information on the control plant, and
- making decisions concerning the control plant.

The controlling unit is a combination of the information unit and the decision unit. Each self-controlling enterprise process

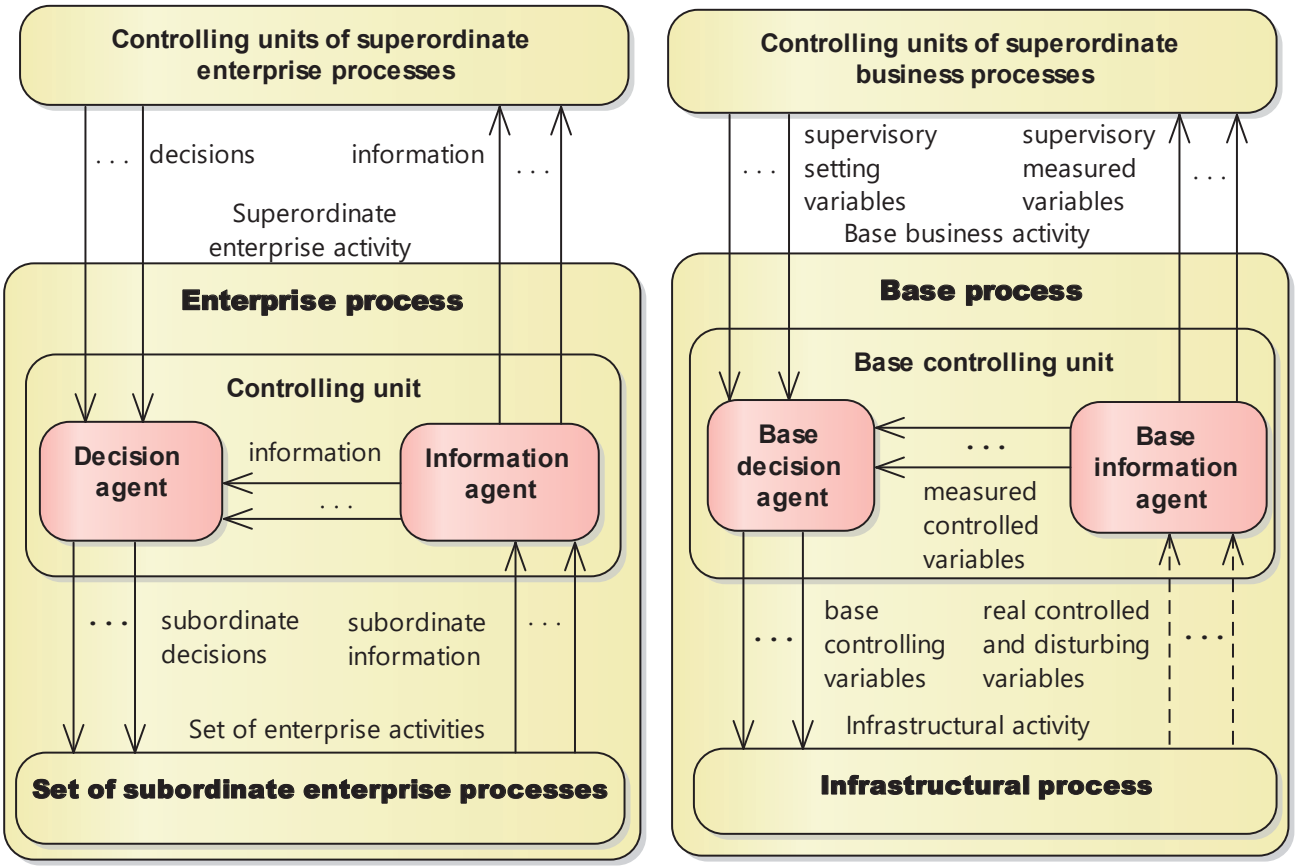

Fig. 3. Enterprise processes as control systems

has exactly one controlling unit. Information units and decision units are functional units of enterprise processes. Each functional unit includes its functional agent, $k \in Kf$, designed for data processing, and its own memory. Functional agents are information and decision agents,

$$Kf = Kfi \cup Kfd, \qquad Kfi \cap Kfd = \emptyset.$$

In the case of self-controlling business processes, the information unit represents processing information, which is derived from subordinate processes (Fig. 3). For both base and business processes, it represents also the first part of the control algorithm (Fig. 2.5 in [4]), which introduces its input variables to the memory. The decision unit is a location for the main part of the control algorithm.

Any functional agent is a particular case of a business agent, $k \in Kf \subset Kb$, that is defined as an ordered set of business transitions, $k \in K \subset Kb$, i.e. elementary software objects designed for processing information and decisions. Each business transition belongs to a definite functional agent, and a definite controlling unit. Thus, it is a component of a definite enterprise process and a definite enterprise activity,

$$p(k) \in P \subseteq A, \quad for \ k \in K.$$

The output variables of functional units are referred to as functional variables, $i \in I$. The output variables of information and decision units are named, respectively, information variables, $i \in Ii$, and decision variables, $i \in Id$:

$$i \in I = Ii \cup Id, \qquad Ii \cap Id = \emptyset.$$

Business transitions belonging to information and decision agents are referred to as information transitions, $k \in Ki \subset K$, and decision transitions, $k \in Kd \subset K$. Their output variables are the same information and decision variables that are output variables of functional agents.

The base controlling variables affect infrastructural control plants through their actuating devices. Decision variables in this case are supervisory setting variables. Moreover, information variables are measured controlled variables and measured disturbances.

2.2. Instants of discrete time. EntPC systems are multilevel discrete-time control systems. This means that information processing is allowed only at discrete-time instants that are separated by discrete-time periods, whose length depends on the organizational level. For any given discrete-time instant, reports (information) are processed first, and then, plans and orders

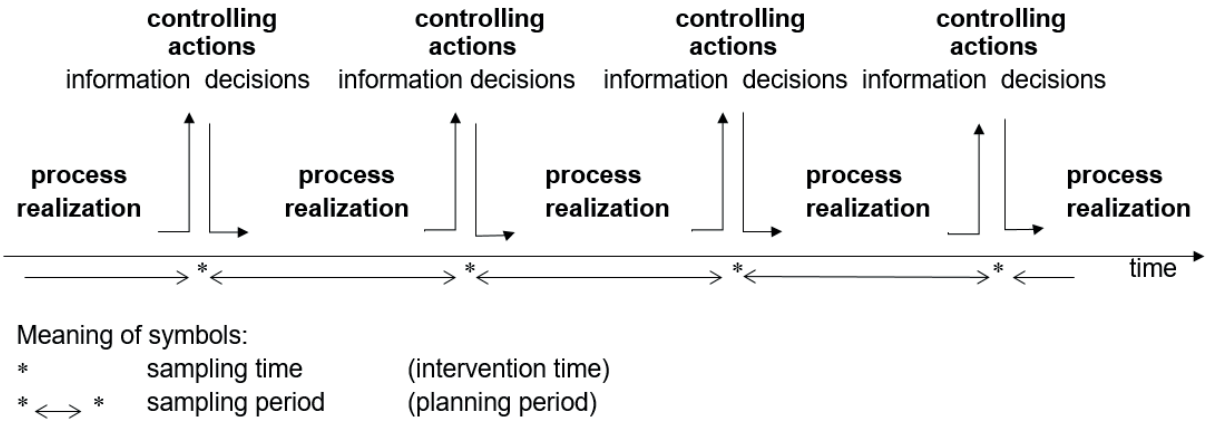

Fig. 4. Discrete-time instants

(decisions) are made (Fig. 4). Discrete-time periods and their final instants are identified by the following pairs:

$$(l, t) \in Tl \subset L \times T,$$

in which the identification numbers of time instants obtain integer values, $t \in T$, from the sets attributed to their time scale numbers, $l \in L$,

$$t_l^- \leq t \leq t_l^+, \quad for \ \ l \in L.$$

In management systems, discrete-time periods are often referred to as planning periods. In direct control systems, they are named sampling periods.

In control systems, the variables attributed to a specific moment in time are often referred to as signals [4]. Hence, the values $y_i(l, t)$ of functional variables, $i \in I$, recorded at discrete-time instants, $(l, t) \in Tl$, are values of the signals of functional variables,

$$(i, l, t) \in It \subset I \times L \times T.$$

In EntPC systems, no business activity on a given organizational level, $l \in L$, can be of execution duration shorter than the length of the corresponding discrete-time period. In contemporary automatic control systems, at the level of direct control of manufacturing infrastructural processes, sampling periods are shorter than one second. In management systems, planning periods depend on the organizational level of the enterprise. Higher levels correspond to longer planning periods. For example, discrete-time periods may be determined as follows:

$l = 4$ 1 month for enterprises and for organizational systems of their environments,
$l = 3$ 1 day (and night) for work sites of enterprises and their departments,
$l = 2$ 1 hour for workshops and their work centers, and
$l = 1$ 0.1 second for workstations and their base processes.

2.3. Business transitions and business events. In EntPC systems, information flows consist in recording values of variables determined by business transitions in memory places in the controlling units of individual enterprise processes and then in reading them by other business transitions (Fig. 5). Information variables are remembered in the same controlling unit that includes information transitions which record them. In contrast, the decision variables are kept in the controlling units that include the decision transitions that read their values. To clarify, decisions are remembered where they are to be executed. However, the reported decision variables, as inputs to the superordinate decision agents or to decision agents of receiving processes, are information variables.

Couplings between functional units of different enterprise processes have been briefly described in [10]. In a general case, a given enterprise process may have multiple superordinate processes (including business systems and their roles), multiple subordinate activities, multiple receiving processes and multiple delivery processes. Hence, the controlling unit of an enterprise process (Fig. 5) may have couplings with many superordinate controlling units, with many controlling units of subordinate activities as well as with many controlling units of delivery and receiving processes.

In EntPC systems, business transitions are the only software objects that perform data processing operations. Some business transitions calculate values of Boolean functions but in a general case they can perform the more complex procedures of data processing, e.g. the algorithm of digital PID controllers in direct control systems [22], the MRP algorithm [23] for ERP systems and the like.

Every business transition has exactly one business action, i.e. an operation that processes its input variables into output variables. Additionally, it has one operation for shifting its output state variables, which is executed when the clock of a given time scale initiates the next discrete-time period. It also has, like all other EntPC system objects, one operation for reading input variables and one operation for recording output variables.

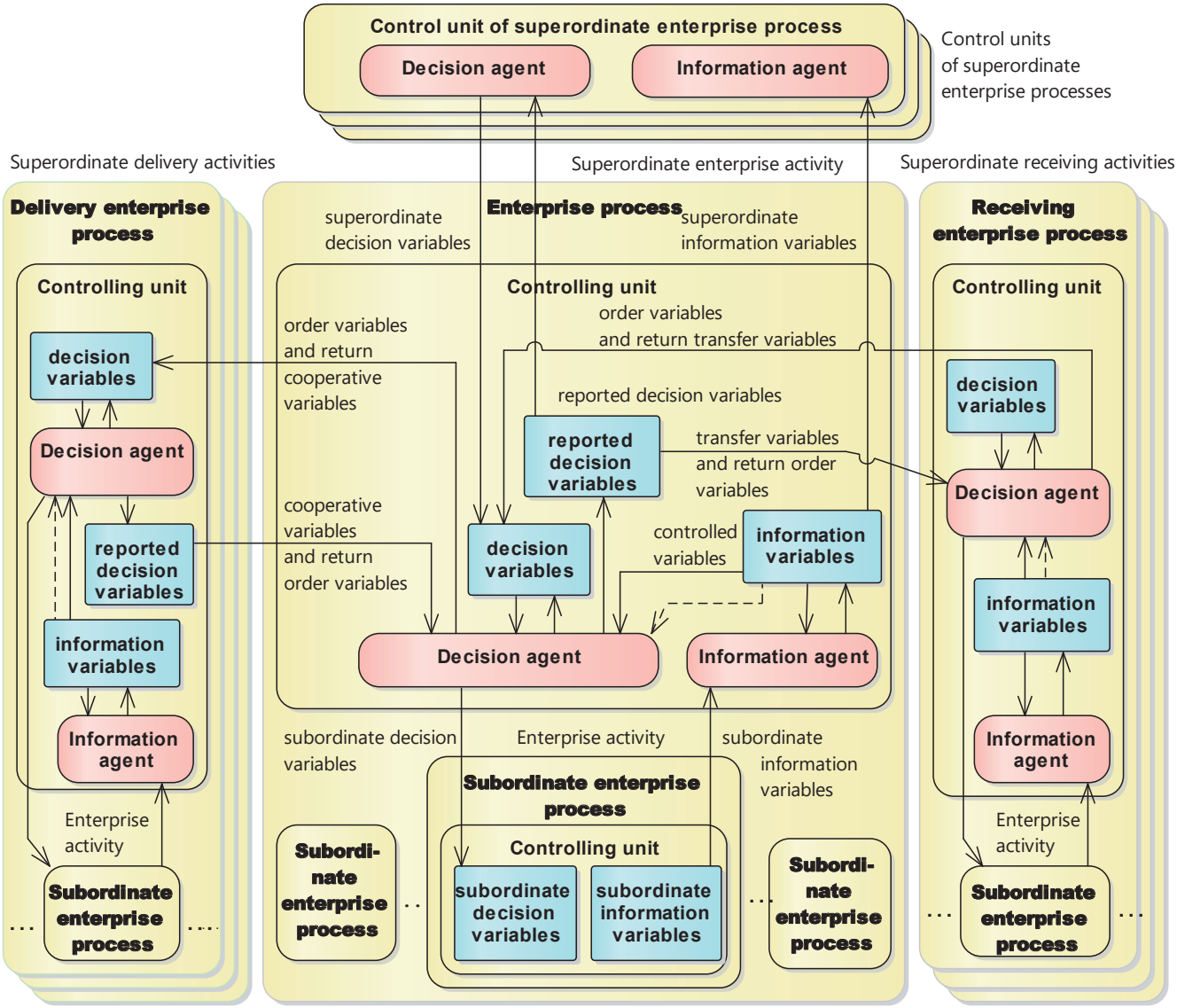

Fig. 5. Information flow between functional agents of enterprise process

The main input and output variables of business transitions are functional variables and state variables related to them. The others are guard conditions, $j \in Jg$, that are binary variables used for controlling execution of transitions. Guard variables, $i \in Ig \subset I$, as functional variables (Fig. 6), are attributes of structural objects, whereas guard conditions are directly attributed to business transitions.

Business transitions are divided into guard transitions, $k \in Kg \subset K$, that process only guard variables as well as guard conditions, and procedural transitions, $k \in Kp \subset K$, that process also other state variables,

$$k \in K = Kp \cup Kg, \qquad Kp \cap Kg = \emptyset .$$

Guard transitions correspond to events and gateways of the BPMN standard [13]. Cause-effect dependencies between their input and output variables are Boolean functions. In the case of procedural transitions, procedures of business actions are performed by corresponding elements of the implementation environment [20].

Every business agent is a location for the corresponding information-decision process, $k \in Pid = Kb$, that is defined as an ordered set of the business actions, $k \in K$, which can be executed at the same discrete-time instant. Business actions and i-d processes have the same identifiers as corresponding business transitions and business agents, respectively. An individual business action is a special case of an i-d process.

Business transitions stimulated by clock impulses at consecutive discrete-time instants investigate the states of passive objects in their environment to decide whether to begin their actions. In this sense, they are autonomous software objects [24]. Business transitions are coupled with the others by functional

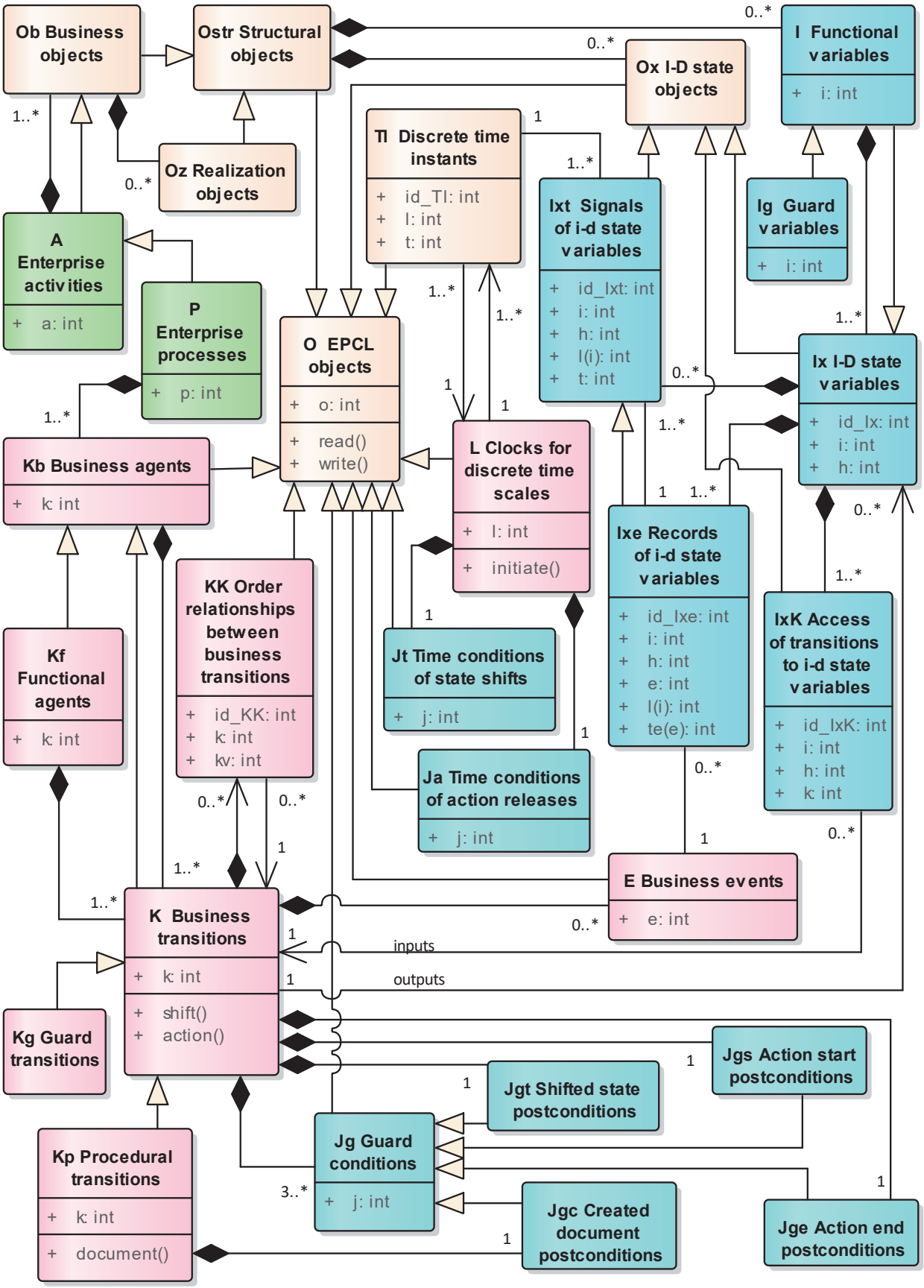

Fig. 6. The class diagram for discrete time, business transitions and i-d state variables

variables and guard conditions. Consequently, EntPC systems may be counted among multi-agent control systems with passive interactions between agents [25].

A business event, $e \in E$, is an action execution of a definite business transition:

$$k = k(e) \in K, \quad for \ \ e \in E.$$

The duration of each business action is formally equal to 0, and the entire action is attributed to a concrete discrete-time instant. If the actual duration of the procedural action may be neglected, then the action calls the corresponding element of the implementation environment directly. Or, in another case, the business action of the procedural transition calls the corresponding business activity,

$$a = a^m(k) \in Am \subset A, \quad for \ \ k \in Kpm \subset K,$$

of the managerial business process,

$$p = p^m(k) \in Pm \subset P, \quad for \ \ k \in Pidm,$$

that corresponds to the i-d process encompassing this action. The duration of the managerial activity that is performed at the lower organizational level is greater than 0, but it may be neglected at the level of the procedural transition. In the case of a guard transition, $k \in Kg$, its business action is performed by the transition itself and its duration is imperceptible.

Managerial processes are workflow business processes, whose input and output products are documents [8]. Hence, the procedural business transition includes an additional operation for creating the output document (Fig. 6). This operation should be executed before the main action of this transition.

3. Information-decision state

3.1. Definitions. One of the typical problems of enterprise process management is planning current activities on the grounds of their planned future effects and forecast disturbances. Therefore, the information-decision state for an EntPC system is defined as a set of values of i-d state variables that represent all current and past information as well as forecasts and decisions concerning the future, which are recorded in the memory of the entire controlling system and are needed in order to make new decisions. The values

$$x_{ih}(l, t)$$

of i-d state variables

$$(i, h) \in Ix \subset I \times H$$

at the current discrete-time instants (l, t) are assigned to the instants $(l, t + h)$, shifted in time, back or forward, by a definite number, $h \in H$, of discrete-time periods.

To compare the i-d state with the state of a controlling unit in the general model (1)(2), defined as per the control theory, one can try to apply this model to a functional unit of a self-controlling enterprise process (section 2.1). Equations (1)(2) would be the same as for a controlling unit. The algorithm of the information unit would be executed for the period n at the end of this period, i.e. at instant n. Next, in the same instant, regarded as the beginning of the $n + 1$ period, the algorithm of the decision unit would be performed.

Generalizing the linear equation of the 'input-output' model of the controlling unit, which is presented on page 33 in [4], one can write:

$$v_n = \varphi(v_{n-1} \ldots v_{n-m}, u_{n-1} \ldots u_{n-m}, u_n).$$

On the other hand, however:

$$v_n = \eta(\boldsymbol{x}_n) = \eta\big(\boldsymbol{f}(\boldsymbol{x}_{n-1}, u_n)\big).$$

Hence:

$$\boldsymbol{x}_{n-1} = \big[v_{n-1} \ldots v_{n-m}, u_{n-1} \ldots u_{n-m}\big] = \big[\boldsymbol{v}_{n-1}, \boldsymbol{u}_{n-1}\big]. \quad (3)$$

This means that the state vector of the controlling unit may be constructed as a vector whose components are values of their output and input variables, shifted in time back by a definite number of discrete time periods. The values of the current state variables are calculated on the grounds of the previous state, as the following expressions:

$$x_{n-h+1} := x_{n-h}, \quad for \ \ h = m \ldots 1 \quad (4)$$

$$x_n = \big[v_n, u_n\big] := \big[\varphi_n(\boldsymbol{x}_{n-1}, u_n), u_n\big]. \quad (5)$$

The (4)(5) model is adequate for information units in EntPC systems, but it is not appropriate for describing the influence of future values of decision input variables:

$$\boldsymbol{u}_n^+ = \big[(u_{n+h} \big| h = 0, 1, \ldots h^+)\big]. \quad (6)$$

Hence, for functional units of EntPC systems it should be substituted by the following formulas:

$$x_{n+h} = \big[v_{n+h}, u_{n+h}\big], \quad for \ \ h = h^- \ldots 0 \ldots h^+, \quad (7)$$

$$x_{n+h} := x_{n+h+1}, \quad for \ \ h = h^- \ldots -1, \quad (8)$$

$$x_{n+h} := \big[\varphi_{n+h}\big(\boldsymbol{x}_{n-1}, \boldsymbol{u}_n^+\big), \boldsymbol{u}_n^+\big], \quad for \ \ h = 0, 1, \ldots h^+. \quad (9)$$

The controlling unit of any enterprise process is composed of two functional agents and memory place sets containing i-d state variables (Fig. 5). These functional agents have couplings, through i-d state variables, with multiple functional agents of subordinate, superordinate and cooperating enterprise processes. On the other hand, the entire controlling system of an EntPC system is a network of controlling units of its enterprise processes (Fig. 1). So, it is a network of functional agents and

memory places of i-d state variables that separate these agents. It is also a network of business transitions and i-d state variables that are processed by them, because every functional agent is an ordered set of business transitions.

The i-d state variables are inputs and outputs of business transitions. One should notice that all of them are output variables of algorithms performed in definite business transitions of the whole controlling system. In this sense, every EntPC system is a closed system. Its i-d state variables, i.e. components of the i-d state vector

$$\boldsymbol{x}(l,t) = \left[\boldsymbol{x}^{v}(l,t), \boldsymbol{x}^{u}(l,t)\right] \tag{10}$$

may be counted among:

- **external input variables**, whose values,

$$\boldsymbol{x}^{u}(l,t) = \boldsymbol{f}^{ext}(l,t), \quad for \ \ (l,t) \in Tl, \tag{11}$$

  are introduced to the controlling system from the outside, by its users and by measurement devices, at the end of a discrete-time period, and
- **internal i-d state variables**, whose values,

$$\begin{aligned} &\boldsymbol{x}^{v}(l,t) = \boldsymbol{f}^{int}\big((l,t), \boldsymbol{x}(l,t-1), \boldsymbol{x}^{u}(l,t)\big) \\ &for \ \ (l,t) \in Tl, \end{aligned} \tag{12}$$

  at the end of discrete-time instants are calculated on the grounds of values of i-d state variables from the end of the preceding period. Direct dependency on time, (l,t), indicates that in a general case the authorized employees of the enterprise can correct the results of the procedure.

The values of i-d state variables coming from both sources should be sustained till the final instant of a discrete-time period.

These values at discrete-time instants (l,t), as values of signals of i-d state variables,

$$(i,h,l,t) \in Ixt \subset I \times H \times L \times T,$$

are equal to the values of functional variable signals that are shifted in time (section 2.2):

$$\begin{aligned} &x_{ih}(l,t) = y_i(l,t+h), \\ &for \ \ h_i^- \le h \le h_i^+, \ \ i \in I \ \ (l,t) \in Tl. \end{aligned} \tag{13}$$

Conversely, the value of a functional variable is equal to the value of the i-d state variable with a zero-time shift:

$$\begin{aligned} &y_i(l,t) = x_{ih}(l,t) \,|\, h = 0 \wedge (i,h) \in Ix, \\ &for \ \ t_l^- \le t \le t_l^+, \ \ l \in L \ \ i \in I. \end{aligned} \tag{14}$$

Thus, in the memory of an EntPC controlling system, functional variables, $i \in I$, may be represented by corresponding i-d state variables,

$$(i,h) \in Ix \,|\, h = 0.$$

One functional variable may correspond to many i-d state variables. I-d state variables, like functional variables, are divided into information state variables, $(i,h) \in Ixi \subset Ix$, and decision state variables, $(i,h) \in Ixd \subset Ix$,

$$i \in Ix = Ixi \cup Ixd, \qquad Ixi \cap Ixd = \emptyset.$$

Business events, which are regarded as executions of business transitions, may insert the records of i-d state variables,

$$(i,h,e) \in Ixe \subset Ix \times E,$$

into the system memory. Each record (i,h,e) of an i-d state variable (i,h) is also an effect of one definite event, $e \in E$, and is a formal component of this variable. This record corresponds to the i-d state variable signal at the instant of its creation and, potentially, to the signals at certain future time instants.

From the IT point of view, equation (12) is a static model of the cause-effect dependencies between input and output variables of procedures that are performed at settled discrete-time instants. The reason being that business transitions are static objects (see the first paragraph of section 4.1). However, from the control theory perspective, it is also a dynamic model of the EntPC controlling systems, because coordinates of the i-d state vector $\boldsymbol{x}(l,t)$ represent the state of functional variables at current instants and at those shifted in time.

3.2. Information-decision state in the EntPCL metamodel. Practical conclusions from analysis of EntPC systems, regarded as complex control systems, always concern i-d state variables perceived as attributes of enterprise processes or attributes of structural objects that belong to these processes. Hence, the conceptual model of the structure of enterprise processes [2, 10, 15] is one of the main subjects of EntPC theory. Relationships between concepts corresponding to all the sets of software objects in any EntPC system are presented as relations between classes in class diagrams of the EntPCL metamodel. The Enterprise Process Control Language (EntPCL) is the graphical language designed for modeling structures of concrete enterprise processes. Its object diagrams and class diagrams of its metamodel are simplified UML diagrams [26]. Similar metamodels exist for graphical languages designed for modeling enterprise architectures. They include e.g. ArchiMate [27] and UEML [28, 29], but areas of facts modelled in these languages are different.

Class diagrams describing relationships between the most important subclasses of structural objects and between structural objects and information-decision state variables of EntPC systems have been discussed in [2, 10, 15]. Further details of relationships between subclasses of functional variables and i-d state variables are shown in Fig. 6.

The EntPCL metamodel encompasses all structural objects of EntPC systems and all variables that are their attributes as well as all associative objects that represent their relationships. Structural objects

$$o \in Ostr = Ob \cup Oz$$

are divided into business objects, $o \in Ob$, and realization objects, $o \in Oz$.

Changeable attributes of business objects are referred to as business variables, $i \in Ib$. Realization variables, $i \in Iz$, are associative objects that represent associations of business variables with realization objects. Both business and realization variables are functional variables,

$$i \in I = Ib \cup Iz, \qquad Ib \cap Iz = \emptyset .$$

Formally, business variables, realization variables and functional variables are components of business objects, realization objects and structural objects, respectively. What is more, any i-d state variable is a component of a functional variable and, indirectly, a component of a specific business object and a specific enterprise activity (Fig. 6),

$$a(i, h) \in A, \quad for \ \ (i, h) \in Ix .$$

The set of functional variables includes the following:

- quantity variables, e.g. the quantities and flow rates of products, production costs of specific products, income of an enterprise, etc.;
- quality variables, e.g. length, diameter, color, temperature, etc.;
- time variables, e.g. the due date of a business task;
- existential variables – i.e. binary variables that indicate whether specific business objects exist;
- and guard variables, i.e. binary functional variables that are used to control the execution of business transition actions (section 2.3).

The classes of EntPCL objects that are discussed in this study are as follows:

A	enterprise activities
Aa	business activities
Am	managerial business activities
Dm	delivery managerial activities
E	business events
Ee	end events
Es	start events
H	numbers of shift periods identifying i-d state variables
G	business roles
I	functional variables
Ib	business variables
Id	decision variables
Ig	guard variables
Ii	information variables
IK	inputs of functional variables to business transitions
It	signals of functional variables5
Ix	i-d state variables
Ixd	decision state variables
Ixe	records of i-d state variables
Ixi	information state variables
IxK	inputs of i-d state variables to business transitions
IxKp	inputs of i-d state variables to procedural transitions
Ixt	signals of i-d state variables
Ixu	external input i-d state variables
Ixv	internal i-d state variables
Ixva	internal i-d state variables from automatic actions
Iz	realization variables
Ja	time conditions of action releases
Jg	guard conditions
Jgc	‘created document’ postconditions
Jge	end postconditions
Jgs	start postconditions
Jgt	shifted state postconditions
Jt	time conditions of state shifts
K	business transitions, business actions
Kae	end transitions of enterprise activities
Kame	end transitions of managerial activities
Kams	start transitions of managerial activities
Kas	start transitions of enterprise activities
Kb	business agents
Kf	functional agents
Kfd	decision agents
Kfi	information agents
Kd	decision transitions
Ke	end transitions of enterprise processes
Kg	guard transitions
Ki	information transitions
KK	order relationships between business transitions
KKp	order relationships of procedural transitions
Kme	end transitions of managerial processes
Kms	start transitions of managerial processes
Kp	procedural transitions
Kpm	substituted procedural transitions
Ks	start transitions of enterprise processes
Kv	business transitions preceding other business transitions
L	time scale numbers, clocks for discrete-time scales
Ostr	structural objects
Ob	business objects
Oz	realization objects
P	enterprise processes
Pid	information-decision processes
Pidm	substituted information-decision processes
Pm	managerial business processes
R	business products
Rinm	input documents of managerial activities
Rmm	main documents of managerial activities
Tl	discrete-time periods
U	business units
W	business works

4. Data processing in controlling units of enterprise processes

4.1. Input and output variables of business transitions. Any coherent fragment of an EntPC controlling system may be modelled, with the use of an EntPCL object diagram, as a network of business transitions and the memory places of i-d state variables that separate them (Fig. 7). Their structure is similar to

the Organizational Information-Transition Nets (OITN) [30]. The OITN, based on the CPN Tools [31], have been used for simulation research on the theory of enterprise resource control (ERC) that was the early version of EntPC theory. Disregarding the guard conditions, one can say that business transitions, like transitions in colored Petri nets, are static objects that have no state. Nonetheless, the controlling units of enterprise processes, as networks of business transitions and i-d state variables, are dynamic systems, because at specific time instants the values of the output variables of transitions depend on the values of their input variables at instants shifted in time.

Business transitions lack memory, so all information concerning the state of EntPC systems is placed in memory places between transitions (Fig. 7). This is an important feature of these systems. In particular, it is also the reason why the memory of decisions concerning enterprise processes is attributed to these processes rather than to the processes during which the decisions were made. For example, the decision variable, $(i, h) = (4, 2)$, of the transition $k = 4$, is not remembered in the controlling unit of the process to which this transition belongs but in the controlling unit of another process that includes transitions $k = 6$ and $k = 7$.

Although business transitions are static objects, the functional block diagrams with dynamic functional blocks may always be substituted by corresponding networks of business transitions and i-d state variables. However, the memory places belonging to specific functional blocks of definite functional units may be attributed to different functional units. E.g. the block diagram presented in Fig. 8 is substituted by the EntPCL object diagram from Fig. 7, in which the memory place St42 from the block containing transition tr4 is moved to the decision unit containing transitions DecTr6 and DecTr7.

A business transition may have many output variables, but each functional variable and each i-d state variable is an output variable of one specific business transition,

$$\begin{aligned} k(i) &= k^{rec}(i) \in K, \quad for \;\; i \in I, \\ k(i,h) &= k^{rec}(i) \in K, \quad for \;\; (i,h) \in Ix, \;\; i \in I, \end{aligned} \tag{15}$$

which can record its values. This function dependency constitutes one of EntPC theory axioms. This means that if two functional variables of the same type are outputs of two different business transitions then they are different variables, even if they are attributes of structural objects of the same type, belonging to the same organizational system. The unambiguous attribution of each i-d state variable of an EntPC system to the transition recording its values may suggest that it is a state

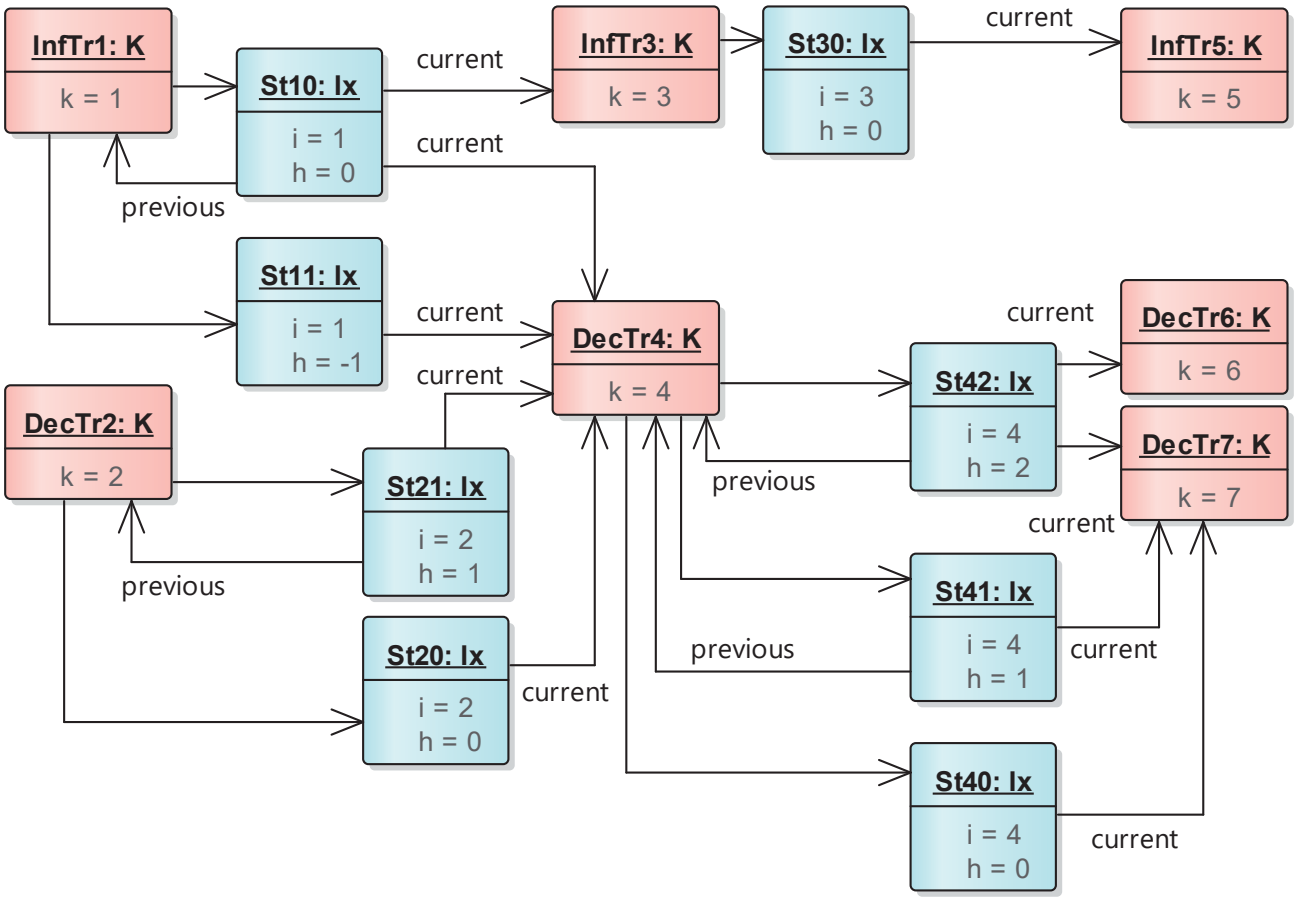

Fig. 7. EntPCL object diagram as an example of processing i-d state variables by business transitions

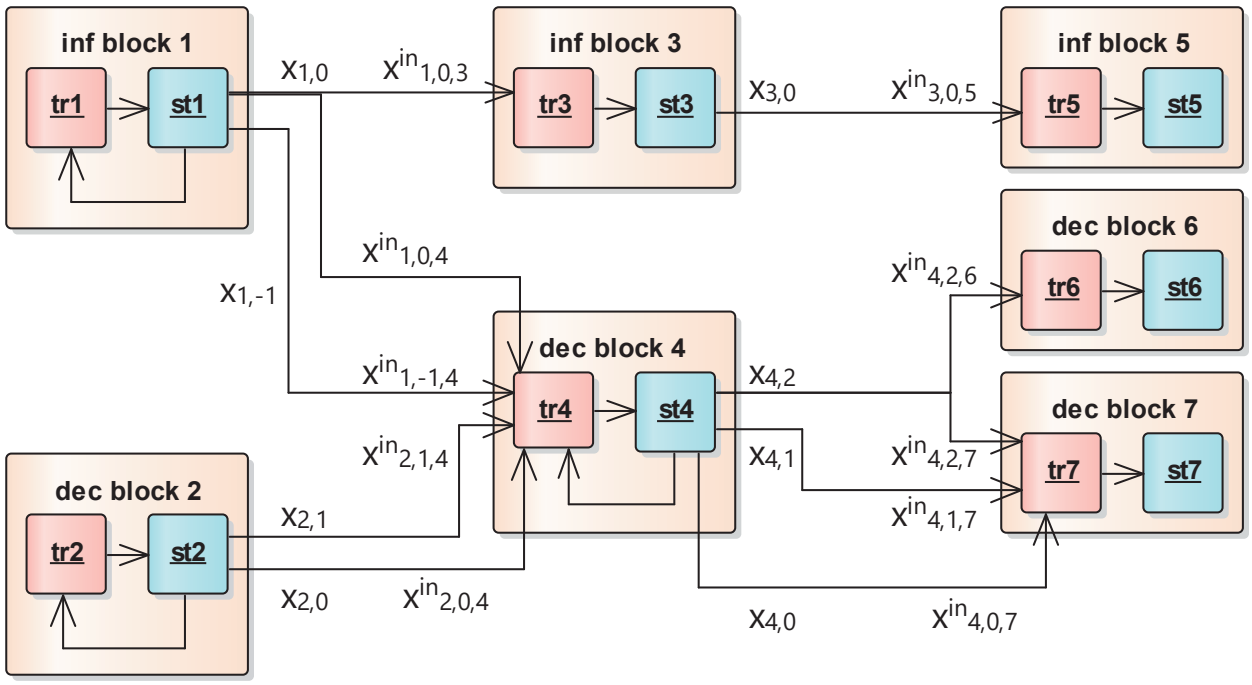

Fig. 8. Block diagram as an example of information flow between control system elements

variable of this transition. However, this is not the case. I-d state variables are remembered in the controlling units of enterprise processes and are components of these processes, but they do not belong to business transitions or functional agents of the processes (Fig. 5).

Access of business transitions to i-d state variables is represented by the following associations:

$$(j, b, k) \in IxK \subset Ix \times K,$$

that indicate the transitions which may read the values of individual variables. These associations are referred to as inputs of i-d state variables because they identify the values,

$$\left(x_{j,b}(l, t) \middle| (j, b, k) \in IxK\right) \tag{16}$$

of input variables, $(j, b) \in Ix$, which may be read by business transitions, $k \in K$, at discrete time instants, $(l, t) \in Tl$. Each input of an i-d state variable corresponds to exactly one input of a functional variable to the business transition,

$$(j, k) \in IK \subset I \times K.$$

4.2. Procedures of i-d state processing. The input data of procedures calculating current values of internal i-d state variables are not only their previous values and the current values of the external inputs but also the current values of those internal i-d state variables which have already been calculated at the same discrete time instant (Fig. 7). Therefore, to achieve an effect in the form of the (10) (11) (12) model, the i-d state should be shifted in time before calculations by one discrete-time period, and formula (12) should be properly modified.

Moving to a new discrete time instant does not change the values of i-d state variables but does change their identifiers, i.e. it shifts the state towards earlier periods. Hence, immediately after creating (by means of a clock) the initial instant of a current discrete-time period,

$$(l, t) := (l, t + 1), \quad for \ \ (l, t) \in Tl, \tag{17}$$

and prior to generating current information on the i-d state, one should decrease the values of the time shifts of i-d state variables by 1 relative to the current time instant, as follows:

$$\begin{aligned} &x_{i,h}(l, t - 1) := x_{i,h+1}(l, t - 1), \\ &for \ \ i \in I, \ \ h_l^- \le h \le h_l^+ - 1, \ \ (l, t) \in Tl. \end{aligned} \tag{18}$$

Then, the tentative values of internal i-d state variables should be set for the current period and its final time instant:

$$\begin{aligned} &x^v_{i,h}(l, t) := x^v_{i,h}(l, t - 1), \\ &for \ \ (i, h) \in Ixv, \ \ (l, t) \in Tl. \end{aligned} \tag{19}$$

If the transition computing a given i-d state variable does not act at the initial instant of the current period, then the value resulting from substitution (19) remains until the end of this period. If the transition does act, then this value is regarded as the value of the input variable, which represents the previous i-d state of the system.

Procedures calculating individual decision variables should act at the beginning of the current discrete-time period (l, t), i.e. at instant $(l, t-1)$. Procedures whose outputs are individual information variables should be executed at the end of the current period, at instant (l, t). However, in practice, procedures introducing individual external input variables may be performed at the end of a discrete-time period, even if they are decision variables. In such a case, their outputs,

$$x^{u}_{i,h}(l, t) := F^{ext}_{i,h}(l, t), \ for \ (i, h) \in Ixu, \ (l, t) \in Tl, \quad (11b)$$

created in the previous period, are read only at the beginning of the current period. Actions of business transitions that change values of internal i-d state variables,

$$k(i, h) \in K, \quad for \ (i, h) \in Ixv,$$

have no influence on external input variables.

Procedures calculating individual internal i-d state variables are performed as actions of business transitions, $k(i, h) \in K$. The values of output variables of these procedures,

$$x^{v}_{i,h}(l, t) := F_{ih}\Big((l, t), x^{in}_{j,b,k(i,h)}(l, t)\Big|\big(j, b, k(i, h)\big) \in \in IxK\Big), \quad for \ (i, h) \in Ixv, \ (l, t) \in Tl. \quad (12b)$$

depend on the current values of input i-d state variables,

$$\Big(x^{in}_{j,b,k(i,h)}(l, t)\Big|(j, b, k) \in IxK\Big),$$

which are values appearing just before execution of the transition $k(i, h)$. In a general case, because of random sequence of business actions, they may be different from the values at final instant of the current period. Therefore, data processing should be organized in such a way that input i-d state variables are equal to their current values, which are sustained till the end of the period,

$$(j, b, k) \in IxK \Rightarrow x^{in}_{j,b,k}(l, t) = x_{j,b}(l, t), \quad for \ (j, b) \in Ix, \ (l, t) \in Tl. \quad (20)$$

If statement (20) is true, then formula (12b) may be simplified to the following form:

$$x^{v}_{i,h}(l, t) := F_{ih}\Big((l, t), x_{j,b}(l, t)\Big|\big(j, b, k(i, h)\big) \in IxK\Big), \quad for \ (i, h) \in Ixv, \ (l, t) \in Tl. \quad (12c)$$

Formula (12c) is adequate regardless of whether the action of transition $k(i, h)$ is a complex optimization procedure or a calculation of the value of a simple Boolean expression. For automatic business actions,

$$x^{v}_{i,h}(l, t) := F_{ih}\Big(x_{j,b}(l, t)\Big|\big(j, b, k(i, h)\big) \in IxK\Big), \quad for \ (i, h) \in Ixva \subset Ixv, \ (l, t) \in Tl \quad (12d)$$

the values of output variables are automatically committed to their memory places without modifications by EntPC system users.

5. Requirements concerning duration and sequence of business events

5.1. Avoiding transitory states of i-d state variables. Business events have been defined as executions of business actions that are the main operations of business transitions (section 2.3). All business events in EntPC systems are attributed to definite discrete-time instants, but actual durations of action executions are not equal to zero. What is more, for each instant the total effect of these actions depends on their sequence. Consequently, an EntPC system works properly if the following requirements, which are EntPC theory axioms, are satisfied:

- first, the duration of executing a business transition is so short that the interval between the initial moment of the discrete-time period and the end moment of the action is imperceptible relative to the length of this period;
- second, after shifting the i-d state (17,18,19), none of the business transitions can act in a given discrete-time period more than once, to enable attributing one definite current value to a given i-d state variable at a given discrete-time period;
- third, in a given discrete-time instant the business transitions must act according to a definite order, to avoid random variations of i-d state variables.

The first and third requirements will be discussed later (in sections 5.2 and 5.3). The second condition enables avoiding transitory states of i-d state variables. It assures that any i-d state variable may not change at a given instant more than once, because it may change only as a result of an execution of corresponding transition $k(i, h)$. To fulfil this condition, the guard conditions, $j \in Jg$, are used.

At the initial moment of any discrete-time period, all guard conditions (Fig. 6) for a given time scale are set to 0. Just after executing substitution (18), by transition $k \in K$, its shifted state postcondition,

$$j = j^{gt}(k) \in Jgt \subset Jg,$$

obtains the value equal to 1. At the action start moment of a specific transition, the value of its start postcondition,

$$j = j^{gs}(k) \in Jgs \subset Jg,$$

is set to 1. As a result, a business transition cannot act more than once at a given discrete-time instant. On the other hand, each i-d state variable is an output variable of one specific business transition. Therefore, at a given time instant no i-d state variable can change more than once.

Immediately after the end of the action, the value of the end postcondition of the given transition,

$$j = j^{ge}(k) \in Jge \subset Jg,$$

is set to 1. This fact constitutes one of action start conditions for all its subsequent business transitions. The action of a given business transition starts when its start postcondition, $j^{gs}(k) \in Jgs$, is equal to 0, and its initiation function obtains the value of 1. The initiation function of a business transition is a Boolean function of corresponding input guard variables, $i \in Ig$, and of end postconditions, $j \in Jge$, of all its preceding transitions.

EntPC systems are real-time IT systems. Thus, all business actions attributed to a given instant of a given time scale should be finished before the time limit that is settled for this time scale expires. At the first moment in an instant of a specific time scale, $l \in L$, yet before i-d state shifting (18), the value of the time condition of state shifts,

$$j = j^{t}(l) \in Jt,$$

is set to 1. Just after state shifting for output variables for all business transitions on a given organizational level, $l \in L$, the time condition of action releases,

$$j = j^{a}(l) \in Ja,$$

is set to 1. Once all actions attributed to a given period, $(l, t) \in Tl$, are performed, this condition is set to 0, and further actions are forbidden until the next discrete-time instant. Moreover, all guard conditions for all business transitions attributed to a given time scale are set to 0. If certain actions exceed the time limit, then emergency procedures should be launched.

5.2. Sequence of business events. To satisfy the requirement concerning the order of business transitions, the sequence of business action executions in a given EntPC system should be the same for every discrete-time period. Thus, it should be determined by order relationships between business transitions,

$$(v, k) \in KK \subset Kv \times K \subset K \times K.$$

These relationships may be presented in the form of a directed graph (Fig. 9). It is an acyclic graph because no business transition can act more than once at a specific discrete-time instant. To achieve the requested sequence of business events, proper

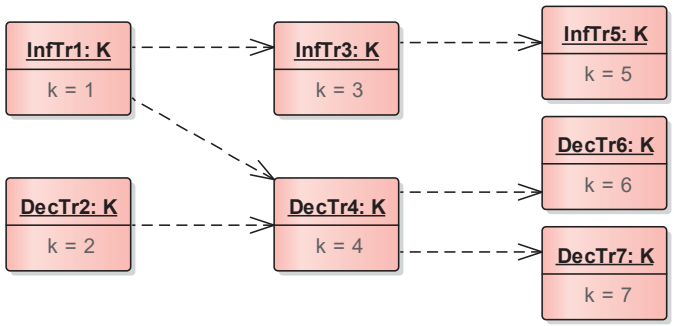

Fig. 9. EntPCL object diagram as an example of order relationships between business transitions

guard transitions, whose action procedures are the initiation functions of procedural business transitions, may be applied.

The sequence of business actions is vital for those procedural transitions that are coupled through current values of i-d state variables (Fig. 7). Therefore, it is required that

$$(j, b, k) \in IxK \;\Leftrightarrow\; \big(k(j, b), k\big) \in KK. \tag{21}$$

This structural requirement must be fulfilled in all EntPC systems as an axiom of EntPC theory. Indeed, the above-discussed axioms guarantee that during data processing at a given discrete-time instant no i-d state variable can change more than once. However, in the case of random sequence of executing business actions, it is not known whether the value of a given input of the i-d state variable to a given business transition,

$$\big(x^{in}_{j,b,k}(l, t) \,\big|\, (j, b, k) \in IxK\big),$$

is 'current' or 'previous'. Axiom (21) ensures that it is the current value,

$$x^{in}_{j,b,k}(l, t) = x_{j,b}(l, t),$$

which will be sustained till the final instant of the current discrete-time period. Thus, statement (20) is true, and formula (12c) is correct.

Axiom (21) imposes certain constraints that concern the action sequence of business transitions belonging to information-decision processes (section 2.3) inside controlling units of enterprise processes as well as constraints concerning couplings between agents belonging to controlling units of different processes.

The i-d processes have the structure of acyclic graphs. This does not mean that in an EntPC system, regarded as a system of elements coupled by cause-effect dependencies, there are no feedback loops. It only means that in such loops certain business events must occur at different discrete-time instants.

Feedback loops might appear for two reasons. Firstly, an output i-d state variable of the end event of a given business work is an input variable to the start event of the next work of the same business process, and as such it influences the value of the same output variable, observed after duration of this next work. Secondly, a feedback loop encompasses business transitions that act at the same discrete-time instant. The simplest way of eliminating such a loop is to substitute it by one business transition with a complex procedure of action. If it is not possible, e.g. in the case of transitions representing participants of negotiations, then one can substitute the set of these transitions by a subordinate managerial business process (section 5.3), whose activities are called by corresponding transitions that belong to the loop.

If the procedural transitions being investigated belong to different functional agents (Fig. 5), then at a given discrete-time instant they should act in accordance with the following rules:

- information transitions of a given enterprise process act after information transitions of subordinate processes,

- decision transitions act after information transitions of the same enterprise process, and after decision transitions of superordinate processes,
- transitions deciding on executions of production orders and taking or rejecting delivery products act after transitions of delivery processes that make their products available,
- transitions deciding on planned production and delivery orders and on declining planned request orders act after transitions of the same process deciding on production orders and taking delivery products as well as after transitions of receiving processes that decide on planned request orders.

Some of these rules have their counterparts in the standard transaction pattern of the DEMO methodology [12].

It should be stressed that decisions on execution of production orders and taking or rejecting delivery products are preceded also by decisions on corresponding planned production and delivery orders, but these relationships are not visible in the structure of i-d processes as they concern decisions made at different discrete-time instants.

5.3. Managerial business processes. The i-d process (section 2.3) is located in the business agent, with the same identifier (Fig. 10). If at least one procedural business action of a given i-d process has the duration comparable with the time-period of the lower level, then this process is substituted by the corresponding managerial business process whose activities are called by actions of the i-d process. Managerial processes are workflow processes, whose input and output products are documents. Managerial activities, which correspond to procedural transitions of i-d processes, are business activities and as such they are perceived as managerial business processes watched from the outside. Therefore, managerial activities have their own input and output documents, that are counted among business products. Input and output i-d state variables of procedural transitions are attributes of input and output business products of corresponding managerial business activities.

It is assumed, for simplicity, that each managerial activity has only one output document, called the 'main document', which is its main business product (Fig. 10),

$$r^{mm}(a) \in Rmm \subset R, \qquad for\ \ a \in Am \subset A.$$

Consequently, input documents of a given managerial activity are main documents of its delivery activities [14],

$$r^{inm}(d, a) = r^{mm}(d) \in Rinm \subset R,$$
$$for\ \ d \in Dm \subset Am,\ \ a \in Am \subset A.$$

Every managerial activity corresponds to exactly one substituted procedural transition,

$$k = k^a(a) \in Kpm \subset Kp \subset K, \qquad for\ \ a \in Am \subset A.$$

Hence, its input and output documents also may be perceived as input and output documents of procedural transitions,

$$r^{mm}(k) \in Rmm \subset R, \qquad for\ \ k \in Kpm \subset K,$$

$$r^{inm}(v, k) = r^{mm}(v) \in Rinm \subset R,$$
$$for\ \ (v, k) \in KK \,|\, (v \in Kv \cap Kpm, k \in Kp).$$

Output i-d state variables:

$$(i, h) \in Ix \,|\, k(i, h) \in Kpm \subset K$$

of a substituted procedural transition, $k \in Kpm$, are recorded as attributes of the main product,

$$r^{mm}(i, h) \in Rmm \subset R,$$

of the substitute managerial activity of this transition,

$$a\big(r^{mm}(i, h)\big) = a^m(k) \in Am \subset A.$$

Input i-d state variables:

$$(j, b) \in Ix \,|\, (j, b, k) \in IxKpm \subset IxK$$

of a substituted procedural transition are read as attributes of the input products of corresponding managerial activity,

$$r^{inm}(j, b) \in Rinm \subset R,$$
$$a\big(r^{inm}(j, b)\big) = a^m(k) \in Am \in A.$$

Moreover, input variables coming from the same preceding business transition, $v \in K$,

$$(j, b) \in Ix \,|\, \big(v(j, b), k\big) \in KKpm \subset KK.$$

are attributed to the same input product,

$$r^{inm}(j, b) = r^{mm}\big(v(j, b)\big) \in Rinm \subset R.$$

According to the BPMN standard [13], executions of business processes are initiated by start events and terminated by end events. In terms of EntPC theory, business process executions are business works, $w \in W$, whereas start and end events,

$$e^s(w) \in Es \subset E,\ \ e^e(w) \in Ee \subset E, \qquad for\ \ w \in W,$$

are executions of start and end transitions of enterprise processes (Fig. 10),

$$k^s(p) \in Ks \subset K,\ \ k^e(p) \in Ke \subset K, \qquad for\ \ p \in P,$$

of managerial processes,

$$k^s(p) \in Kms \subset K,\ \ k^e(p) \in Kme \subset K,$$
$$for\ \ p \in Pm \subset P,$$

of enterprise activities,

$$k^s(a) \in Kas \subset K,\ \ k^e(a) \in Kae \subset K, \qquad for\ \ a \in A,$$

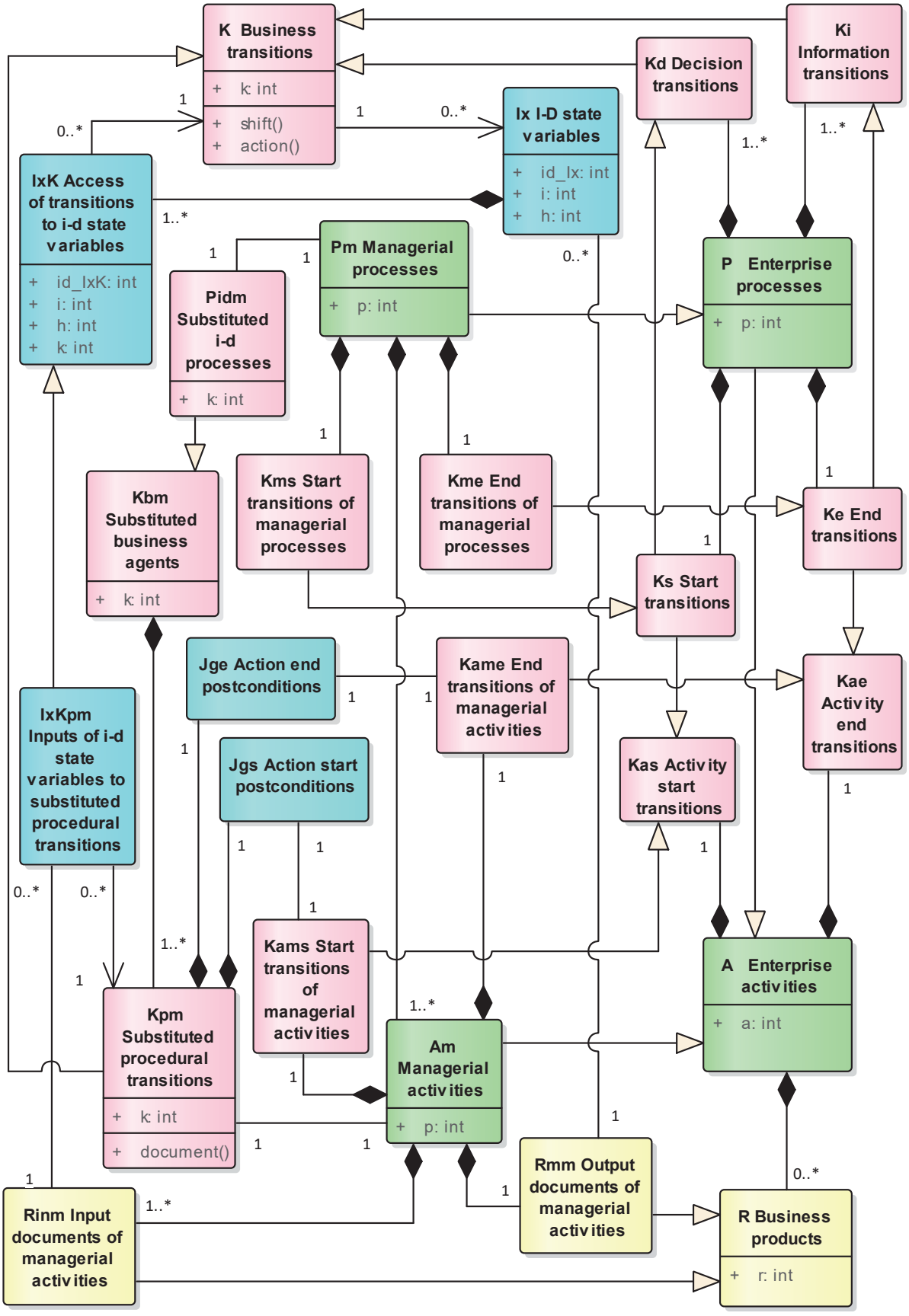

Fig. 10. Class diagram for substituted procedural transitions and managerial processes

and of managerial activities,

$$k^s(a) \in Kams \subset K, \; k^e(a) \in Kame \subset K, \quad for \; a \in Am \subset A.$$

Just after starting the substituted procedural business action, $k \in Kpm$, its start postcondition

$$j = j^{gs}(k) \in Jgs \subset Jg, \qquad for \; k \in Kpm \subset K,$$

is set to 1. This results in reading input i-d state variables and creating input documents of the procedural transition. When the corresponding 'created documents' postcondition

$$j = j^{gc}(k) \in Jgc \subset Jg, \qquad for \; k \in Kpm \subset K,$$

is set to 1, then the action of start transition of the proper managerial activity,

$$k^s\left(a^m(k)\right) \in Kams \subset Kas, \qquad for \; k \in Kpm,$$

is enabled. When the procedure of managerial activity (and procedural action) is terminated, then the action of its end transition,

$$k^e\left(a^m(k)\right) \in Kame \subset Kae, \qquad for \; k \in Kpm,$$

is executed. This event causes the output i-d state variables of the given procedural transition to be recorded in its output document, whereas its end postcondition,

$$j = j^{ge}(k) \in Jge \subset Jg, \qquad for \; k \in Kpm \subset K,$$

is set to 1.

It should be noticed that the start time and end time of a given managerial activity are different discrete-time instants on the organizational level of managerial processes, but on the higher level of production business processes, where corresponding substituted transitions are located, both corresponding events are so close that they may be attributed to the same time instant. If this is not the case, then proper managerial activity should be introduced explicitly to the structure of a given production business process.

6. Conclusions

An enterprise may be perceived as a huge control system with one multilevel controlling system and one control plant, which is a set of infrastructural processes that are control plants of the base direct control systems at the PLC level.

The state of memory places belonging to the controlling system of an enterprise process control (EntPC) system encompasses not only current and past information on its inputs and outputs, but also the future values of input and output variables of all its business transitions. Hence, the information-decision state of the entire EntPC system is defined as a set of i-d state variables that represent all current and past information as well as forecasts and decisions concerning the future. Those are recorded in the memory of the controlling system and are needed to make new decisions. They are quantity and quality attributes of structural objects as well as time variables (e.g. the due date of a business task) and existential variables (i.e. binary variables that indicate whether specific business objects exist in given time periods).

It is shown in the paper that the general mathematical model of an EntPC controlling system may be presented in the form of difference equations with function dependencies between input and output i-d state variables of component business transitions. This model is similar to the model of the controller in classical control theory. In practice, business transitions are executed at the beginning instants of discrete-time periods as elementary procedures of complex IT systems.

The model of enterprise control in the form of difference equations may facilitate transferring the results of classical control theory to the systems of enterprise management, e.g. to analyze enterprise stability and controllability or to assess management quality using criteria and methods applied to the control systems. This is of particular importance for industry 4.0 enterprises, because their management systems should react in real time to the enterprise state changes while, on the other hand, real-time control systems are the subject of control theory. However, to enable fast reaction against local disturbances, the EntPC controlling system does not have just one immediately performed procedure (like in control theory), but instead it is an integrated system of many transitions, whose actions are attributed to the same, consecutive discrete-time instants.

This paper shows how to design complex EntPC systems to ensure that their mathematical models have the same structure of difference equations as the model of a control system with just a single control algorithm. More precisely, it is shown how to ensure that:

- the duration of executing any business transition is so short that the interval between the initial moment of the discrete-time period and the end moment of its action is imperceptible relative to the length of this period;
- none of the business transitions can act in a given discrete-time period more than once, to enable attributing one definite current value to a given i-d state variable at a given discrete-time period;
- in a given discrete-time instant the business transitions act according to a definite order, to avoid random variations of i-d state variables.

Practical conclusions from analysis of EntPC systems regarded as complex control systems always concern i-d state variables perceived as attributes of enterprise processes or attributes of structural objects that belong to these processes. In EntPC theory, an enterprise process is defined as a system of control for a finite, partially ordered set of enterprise activities. Self-controlling of enterprise processes is an essential distinguishing feature of EntPC theory because it enables effective influencing of the course of processes, whereas business process management systems, conformable with the popular YAWL and BPMN standards, can only control launching executions of business activities and monitor their endings.

The main concepts behind EntPC theory (generalized business processes, structural objects, business transitions, i-d state variables and the like) as well as relationships between them are presented in the form similar to the UML class diagrams. These class diagrams belong to the metamodel of the enterprise process control language (EntPCL). On the other hand, EntPCL diagrams, which depict structure of concrete EntPC systems, are patterned on the UML object diagrams.

The class diagrams of the EntPCL metamodel and structural rules of designing EntPC systems may become the starting point for creating the software framework for enterprise process control (SFEntPC). The controlling units of enterprise processes (Fig. 1) will become replaceable building blocks in the software generated in the implementation environment of the SFEntPC. Moreover, business analysts will be able to remove controlling units and embed previously prepared new controlling units without participation of IT engineers. This will obliterate the 'business-IT divide' that refers to the necessity of difficult and prolonged arrangements between business analysts, who understand the actual goals of process reengineering, and IT engineers, who are authorized to make changes to the structure of management systems software [32].

References

[1] M. Zaborowski, "The EPC theory. Basic notions of enterprise process control", *Management and Production Engineering Review*, 1(3), 75–96 (2010).
[2] M. Zaborowski, Introduction to the Theory of Enterprise Process Control, Publ. University of Dabrowa Gornicza, 2016 (in Polish).
[3] B. Scholten, The road to integration. A guide to applying the ISA-95 standard in manufacturing, ISA Publ., Research Triangle Park, NC 27709, 2007.
[4] Z. Bubnicki, Modern Control Theory, Springer-Verlag, Berlin, 2005.
[5] T. Kaczorek, Control Theory, PWN, Warsaw, 1977, (in Polish).
[6] J.H. Blackstone and J.F. Cox, APICS Dictionary. 11th ed., American Production and Inventory Control Society, Alexandria, VA, 2005.
[7] W. van der Aalst and K.M. van Hee, Workflow Management. Models, Methods, and Systems. MIT Press, Cambridge, 2002.
[8] A.H.M. Hofstede, W. van der Aalst, M. Adams, and N. Russell, Modern Business Process Automation: YAWL and Its Support Environment, Springer, Berlin, 2010.
[9] H. Reijers, Design and Control of Workflow Processes. Springer-Verlag, Berlin, 2003.
[10] M. Zaborowski, "Information-decision model for self-controlling enterprise processes", *Engineering Management in Production and Services*, 10 (4), 34–54 (2018).
[11] M.D. Mesarović, D. Macko, and Y. Takahara, Theory of Hierarchical, Multilevel Systems. Academic Press., New York, 1970
[12] J.L.G. Dietz, Enterprise Ontology: Theory and Methodology, Springer-Verlag, Berlin, 2006
[13] M. Weske, Business Process Management: Concepts, Languages, Architectures, Springer-Verlag, Berlin, 2012.
[14] M. Zaborowski, "Tabular models of business process structures", in *Internet in the Information Society*, pp. 281–298, eds. M. Rostański, P. Pikiewicz, and P. Buchwald, Publ. University of Dąbrowa Górnicza, 2015.
[15] M. Zaborowski, "Generalization and composition relationships between objects of enterprise process control systems", in *Internet in the Information Society*, pp. 405–418, eds. M. Rostański, P. Pikiewicz, P. Buchwald, and K. Mączka, Publ. University of Dąbrowa Gornicza, 2016.
[16] H. Kagermann, W. Wahlster, and J. Helbig, Recommendations for implementing the strategic initiative Industrie 4.0, Final report of the Industrie 4.0 Working Group, Acatech, 2013.
[17] L. Monostori, "Cyber-physical production systems: Roots from manufacturing science and technology", *Automatisierungstechnik*, 63(10), 766–776 (2015).
[18] Ch.J. Bartodziej, The Concept Industry 4.0. An Empirical Analysis of Technologies and Applications in Production Logistics, Springer Gabler, Berlin 2017.
[19] Y. Liao, F. Deschamps, E. de Freitas Rocha Loures, and L.F. Ramos, "Past, present and future of Industry 4.0 – a systematic literature review and research agenda proposal", *International Journal of Production Research*, vol. 55 (2017).
[20] J.R. Youssef, G. Zacharewicz, D. Chen, and F. Vernadat, "EOS: enterprise operating systems", *International Journal of Production Research*, vol.55 (2017).
[21] J. Klamka, A. Czornik, and M. Niezabitowski, "Stability and controllability of switched systems", *Bull. Pol. Ac.: Tech*. 61(3), 547–555 (2013).
[22] P.W. Murrill, Fundamentals of Process Control Theory, 3rd ed., ISA Publ., 2000.
[23] J. Orlicky, Material Requirements Planning, Mc Graw-Hill, New York, 1975.
[24] P.C. Lockemann, "Agents", in *Multiagent Engineering: Theory and Applications in Enterprises*, pp. 17–34, eds. S. Kirn, O. Herzog, P. Lockemann, and O. Spaniol, Springer-Verlag, Berlin, 2006.
[25] L. Monostori, P. Valckenaers, A. Dolgui, H. Panetto, M. Brdys, and B. C. Csáji, "Cooperative control in production and logistics", *Annual Reviews in Control* 39, 12–29 (2015),
[26] G. Booch, J. Rumbaugh, and I. Jacobson, The Unified Modeling Language. User Guide, Addison-Wesley, Boston, 1999.
[27] M. Iacob, H. Jonkers, M. Lankhorst, E. Proper, and D.A. Quartel, Archimate 2.0 Specification, The Open Group, 2012.
[28] F. Vernadat, "UEML: Towards a unified enterprise modeling language", *International Journal of Production Research* 40 (17), 4309–4321 (2002).
[29] H. Panetto, "Towards a classification framework for interoperability of enterprise applications", *International Journal of Computer Integrated Manufacturing*, 20 (8): 727–740 (2007).
[30] M. Zaborowski, "Outline of the enterprise resource control systems architecture", in Internet – Technical Development and Applications, eds. E. Tkacz and A. Kapczynski, Advances in Intelligent and Soft Computing, vol. 64, Springer, Berlin, Heidelberg, 2009.
[31] K. Jensen, Coloured Petri Nets, Springer, Berlin,1997.
[32] H. Smith and P. Fingar, Business Process Management: The Third Wave, Meghan-Kiffer Press, Tampa, 2003.