Preprints of the 2013 IFAC Conference on Manufacturing Modelling, Management, and Control, Saint Petersburg State University and Saint Petersburg National Research University of Information Technologies, Mechanics, and Optics, Saint Petersburg, Russia, June 19-21, 2013

Dynamic business process management based on the combined control and data networks

Valery Vykhovanets*, Aleksandra Yatsutko**

**Institute of Control Sciences, Moscow, Russia, (e-mail: valery@vykhovanets.ru)*
***Institute of Control Sciences, Moscow, Russia, (e-mail: aleksundra@yatsutko.net)*

Abstract: The approach for dynamic business process management, based on the use of graphical notation with the combined control and data networks, is considered. Due to the explicit instructions of the data stream it is possible to overcome the automatic semantic gap that arises between the graphical and executable model of business processes. This leads to an increase in the quality control of production and technological processes in enterprises.

Keywords: automation, complex systems, automatic process control, block diagrams, flow diagrams, parallelism.

1. INTRODUCTION

Process-oriented approach, which consists in object management as a multi-level set of inter-related processes, is widely used in modern systems of industrial and organizational management. Business process or simply a process considered as a set of interrelated or interacting activities which transforms inputs into outputs to achieve maximum efficiency of production and business, financial and economic activity of the enterprise. Input process is generally outputs of other processes.

In conventional systems, the process structure is given by the diagram, which is a network of processes (units) and relations management (arcs), that define a partial order of the slave processes (Davis, 2005). Data between processes is usually transmitted indirectly - through the attributes of processes through special objects within the process or the processes common to all the data bus. This arrangement with the data historically inherited from the first analysis methodologies, this is due to the fact that the allocation of the flow of data is difficult to implement and methodically and technically (Mark, 1993). The absence of clearly defined by the data flow makes it impossible to automatically generate executable process models, as well as hamper the actualization already running processes.

In general the aim is to improve the quality of the dynamic process management of enterprises through the use of such a model, which is based on an explicit trace control and data between network units. This solves the problem of automatic generation control program to its description in the form of hierarchically organized process diagrams.

The specific objective is to get an instrument that allows building complex hierarchical models that present high layer with top-level concepts as well as a lower one with low-level terms, and to get a software tool to handle these models.

2. PROBLEMS OF MODELING

Modern business process modeling has arisen on the basis of methodology of structural analysis and design of SADT (Structured Analysis and Design Technique) (Brand, 1993) and the standard IDEF0 (Integrated computer-aided manufacturing DEFinition) (Integration Definition For Function Modeling, 1993). These methodologies are designed for high-level description of processes in the functional aspect and consist of diagrams, text, and a glossary with links to each other. Diagrams - the main components of the model. All the functions and interfaces are represented as units (functional blocks) and the arc (Fig. 1). Each node can be decomposed into another diagram. Note that the notation in the SADT and IDEF0 methodology intended to describe only the composition of processes, rather than their sequence, there is also a perception of the complexity of diagrams and the problem of matching multiple interacting processes.

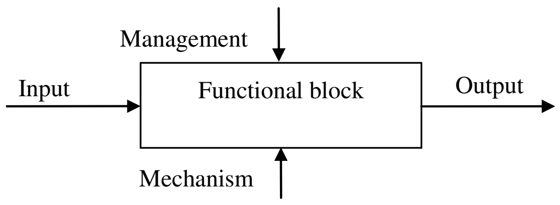

Fig. 1. Functional IDEF0 block.

IDEF0 description of data flows used in notation DFD (Data Flow Diagram) for partial elimination of defects (Gane, Sarson, 1979). The purpose of this presentation - to show how each process converts its input to output. The main components of the data flow diagrams are the internal and external entities, processes, data loggers and data flow (Fig. 2). Each process can be detailed with diagrams or other specifications. Specification languages are generally not standardized and may vary from natural language to visual modeling languages.

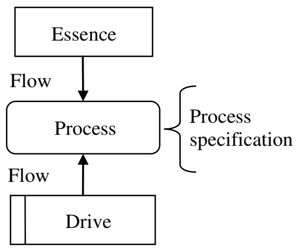

Fig. 2. DFD diagram.

Further development of the methodology of IDEF, namely standards IDEF1 (modeling methodology of information flow within the system) and IDEF1X (methodology for the relational data structures) was intended to address the shortcomings inherent in the model stream.

Note that the model in earlier notations is static by definition and do not reflect the operation of simulated objects in time. The problem of creating dynamic charts trying to solve with IDEF2 (Simulation Model Design) – Dynamic modeling methodology development. But given the complexity of this methodology development of the standard was quickly stopped (Cheremnykh, 2001).

The next step in the modeling process is the development of IDEF3 standard (Mayer, et al., 1995) that is intended to describe the flow of work. IDEF3 models were used to detail the functional blocks IDEF0 with no diagrams of decomposition. Expressive means of standard IDEF3 are close to algorithmic block diagram (Fig. 3).

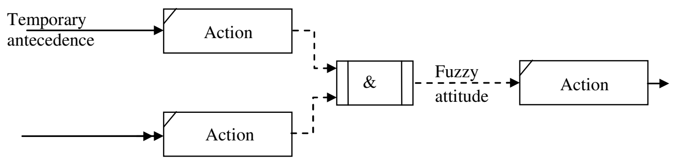

Fig. 3. IDEF3 diagram.

Ambiguous semantics and low expression in the standard IDEF and DFD diagrams led to the unification of the known methods of modeling in the form of the creation of integrated methodologies. One of these methodologies is ARIS (Architecture of Integrated Information Systems) (Davis, 2005). ARIS supports 4 types of models with many (about 80) of species in each type, reflecting different aspects of the studied business processes. For each type of model uses 3 levels of representation: a description of the requirements, project specification and description of the implementation. Both the languages of ARIS own graphical modeling and other languages, in particular, UML (Unified Modeling Language), are used for building models (Booch et al, 2004).

Process model in ARIS is in a sense an extension of IDEF3 - process in ARIS is expressed in the form of diagrams and is a stream of works in series, arranged in order of their performance (Fig. 4).

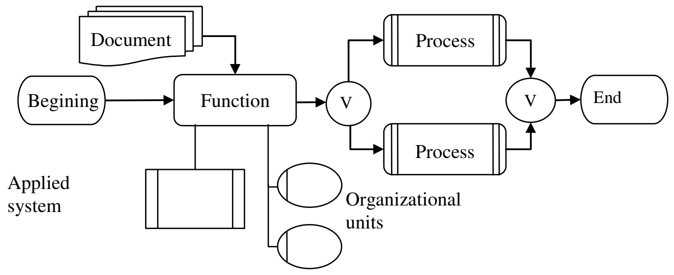

Fig. 4. ARIS process diagram.

Because of the immensity of expressive means the mechanism of methodology filters, that allows you to apply the same process only a certain set of objects and relationships, is embedded in ARIS. The design of such filters requires considerable time and skill of the developer. Ultimately ARIS methodology allows us to describe business processes, analyse the obtained models and to specify the requirements for the implementation of an information system for managing business processes, however, this methodology generates the problem of correct and unambiguous description of the simulated processes.

The necessity to have not only a high level description of processes, but also a full or partial automation has led to the concept of a workflow (Fischer, 2002), in which data and tasks are passed from one performer to another to perform certain actions according to the procedures arch (Fig. 5).

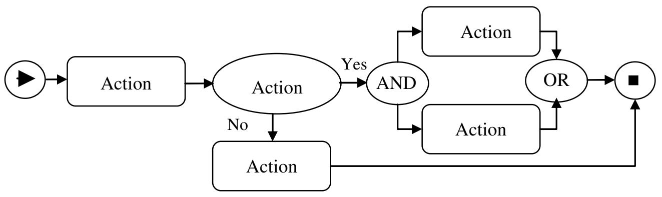

Fig. 5. Workflow diagram.

It should be noted that in the Workflow data is not moved with the management and provides a globally accessible variables and local variables of blocks. This causes considerable difficulties for data synchronizing.

The inflexibility of the models and their inability to provide rapid response to constant changes in the environment are the major shortcomings of methodologies SADT, IDEF, DFD, ARIS, etc., which stimulated the development of the concept of the next generation – BPM (Business Process Management) (Smith, Fingar, 2002) which replaced the radical reengineering comes dynamic control of being able to adjust already automated processes in response to changes in the structure and organization of the company. In this case, modeling tools allow you to create and implement new processes "on the fly", and modeling itself is more like a simulation in Workflow (Fig. 6).

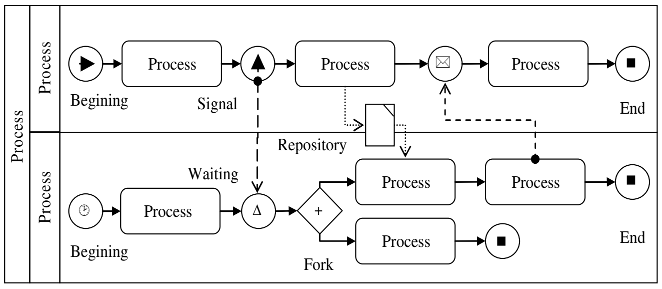

Fig. 6. BPM diagram.

To develop executable models the BPMN notation (Business Process Model and Notation) and the corresponding modeling languages: the language modeling BPML (Business Process Modeling Language), process execution language BPEL (Business Process Execution Language) process and the definition language XPDL (XML Process Definition Language) are used. However, the construction of models in these languages is directly inconvenient for developers because of their syntactic and semantic complexity. In this connection, BPM software developers pay attention to converting graphic process models into executable form. However, the BPEL is mainly used for the automated creation of a template for a "manual tuning". The mass distribution of BPM methodology still lacks technological capabilities that allow you to create and automatically execute the model of automated processes.

Note there are a lot of applications where are used one or more notations or its parts as described above. For example see SCOR – Supply-chain operations reference-model (Poluha, 2007), CIMOSA – Computer-Integrated Manufacturing Open Systems Architecture (Beekman, 1989), MOSYS – software tool for modeling the functional structure, topology, and control rules of systems (Mertins, Rabe, Stiegennroth, 1993), etc.

But the main problem of modern modeling is the presence of a formidable semantic gap between graphics (analytical) and executable process model (Vykhovanets, Iosenkin, 2005). Here, under the semantic gap in understanding the distinction between principles (approach) used for analytical description of the processes and principles (approach), necessary to implement executable models of these processes. Semantic gap is manifested in the fact that the concepts, objects and data structures, which operates the analyst, not the same, and sometimes do not even agree with the concepts, objects, and data structures to be used by the developer of the executable model. To overcome this semantic gap the automatic conversion of the analytical model of the process in its executable form is required.

3. THE COMBINED CONTROL AND DATA NETWORK

The appearance of the problems inherent to modern means of automatic control technology and production processes should be linked to the lack of information which is displayed on the analytical process diagram of fuzzy spatial-temporal structuring and uncoordinated types. To overcome these problems the analytical process model containing not only precisely specifiable connection management processes, but exactly specifiable data sources and ways of transmission, is proposed to use (Yatsutko, 2009a). That is proposed to combine the expressive capabilities of the two models - models to describe workflows and models for data flow.

Graphical notation, based on the combined networks of control and data, allows you to explicitly specify on the same diagram as the flow of control and data flow (Fig. 7).

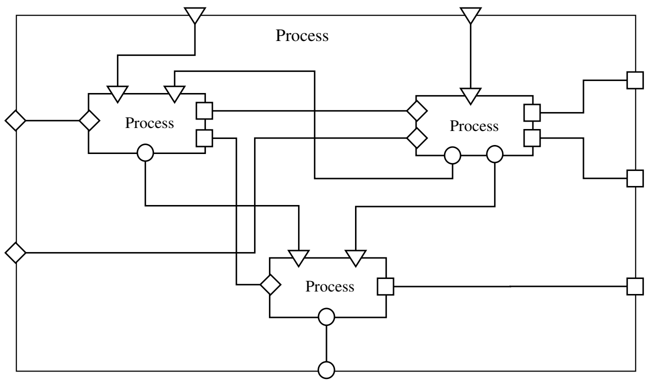

Fig. 7. The combined control and data diagram.

At each unit of the network (shown as a rectangle) there is the control flow and data flow. Control flow entering the unit, called the activator (diamond). Getting activation means the beginning of the unit execution. The control flow output is called an event (square). Event created by one unit can activate one or more others. Data stream is structured input node (triangle). After the unit has executed, the data stream received structured output (circle) that can be transmitted to the next unit as input. When triggered, one or more activators unit is activated, the input data is read and the action attached to the node. Then one or more events and output data are formed.

The unit that do not have a diagram called *base*. The *composite* unit differs from the base one: it is not the last in the hierarchy, it is decomposed into a combined network and it consists of other units, each of which, in turn, can be either basic or composite.

Each basic unit is computational semantics attached to a sequence of actions (operations) in the language of a performer. It can perform a programming system, the virtual machine, the external library, any executive device, and then there is some external interpreter. In turn, the composite unit is constructed from other units, and its computational semantics is uniquely determined by the internal units and their connecting arcs (Yatsutko, 2009b).

When you activate one or more activators of the composite unit it`s internal components are activated in accordance with the given constraints. After the end of the operation of internal units their experiences on the intercom activate other internal components. The functioning of the composite unit is complete when all internal units are no longer activated and completed their functioning. Similar link activation data is transferred between the internal units and the formation of a composite output unit.

Here there is an example to show how this approach can be used. We need to simulate the process of "The elevator algorithm" (it can be included in another process, such as "Automated control of engineering systems of buildings"). In the course of the decomposition we obtain the upper level, which represents the model, suitable for a high level of control, it will be such concepts as "elevator", "floor", "waiting time". And high-level processes: "The launching of the Elevator", "Elevator completion", "Elevator waiting", "Motion on the floor". At the next level of decomposition of the concepts and processes will be others, "Determination of the position of the elevator", "The definition of elevator occupancy", "The determination of total weight of the people in the elevator". While decomposition we get to the lowest level appropriate for the solution of our problem. There will be a notion of the "current supply", "short circuit", "read data from the sensors". The proposed approach allows the software to implement only the latest level of decomposition: no need to programmatically set the logic of movement of the elevator, we only need to program the most common processes. For all other levels of code is generated automatically, which allows to work with the model of behavior of the elevator and change it to the upper levels without the cost of writing new code.

Another example is shown in Fig. 8.

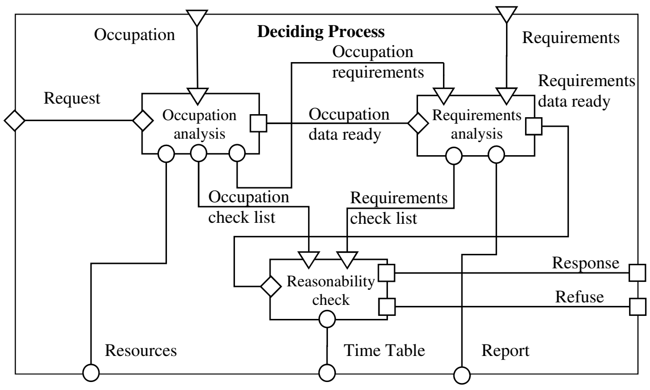

Fig. 8. Some deciding process.

4. THE DIFFERENTIATION AND INTEGRATION OF INPUTS AND OUTPUTS

The interpretation feature of the combined model is the differentiation of activators and input data, and the integration of events and outputs (Yatsutko, 2010a). The process of differentiation activator of the composite unit is to fix the facts activation so that the impact of such activators associated internal components are once and only at the time of activation. The integration process of composite event unit is the accumulation of events, from the internal units at the time of their operation. If a composite unit event was opened at least once, it is the basis for the decision on the transfer of their external host, including the current unit as a component. With the number of events is interpreted as the excitation force activation unit or units, actuators are connected to the event.

Similarly, the process of differentiation of the input data is their fixation at the time of the activation. In turn, the integration of the output is the union of the data on the data output. Integration of the output is done by combining parts of a roll of data obtained after the operation of each indoor unit connected to the data output. The data for such a union represents a named list of values of the form (Name, Value 1, Value 2 ...), where the other values can be a named list. This data structure is convenient to represent a tree, whose root is the output port, and the leaves – data values or other trees (Fig. 9). For example, for the data in Fig. 9 named list is represented as (Output 1, (Name 1, …), (Name 2, Value 2, (Name 3, …)), Value 2, Value 3).

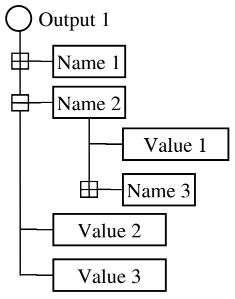

Fig. 9. Data structure.

The above tree structure can be used to specify the interface of input and output nodes. In this case, the design of the site is given the structure of data, the minimum required (directly employed) for its operation. If they do not contain the minimum number during the activation of the node to the input received data, there can be fixed a transmission error that indicates a malfunction of the site - the supplier of the data. In turn, during the construction sites may be issued warnings that made the trace data is not correct, as the output port of the unit (the data provider) does not contain the data necessary for the proper functioning of the unit (the data recipient).

5. COMBINED MODELS EXECUTION

Basic units are executed directly by the executive, which is defined during it`s construction, as it`s computational semantics is given explicitly. Unit execution is the fulfilment of all its internal units. After the differentiation of activators the composite unit activates its internal units and transfer the input data in accordance with the internal routing. Those units that are activated sent for execution. After the end of the operation of each of the activated units the execution process is repeated as long as there are units that were activated in the previous step. In this case, whenever an internal unit has finished the operation, events and output data are integrated.

There are three approaches to the implementation of composite units (Fig. 10) – cyclic interpreter, recursive interpreter, and direct execution (Yatsutko, 2010b).

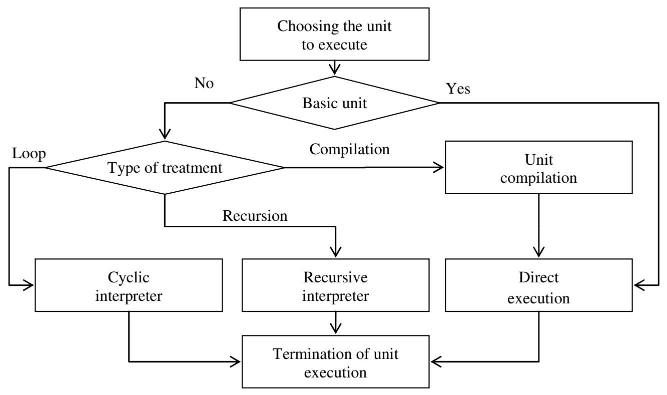

Fig. 10. Combined model unit execution.

When there is a recursive interpreter of the composite unit the direct application of activated basic units is done after determining activated internal units with a recursive call to the same interpreter for each activated composite unit.

Recursive interpretation should be used to quickly executable composite units with low levels of nesting, since the recursive processing at the time of execution of the compound unit is reserved the full amount of computing resources required for the execution of all sub-assemblies.

When there is a cyclical interpretation of the composite unit the descriptions activated internal units are recorded in a special table. Then one or more cyclic interpreters select units from this table according to the priority. The priority is given to those units that are essential to have a greater depth of nesting, or those all component units of which were fulfilled. Cyclical interpretation should be applied in the performance of composite components with a long operation, caused, as a rule, wait for the reaction to some external event.

The third way to implement the execution of composite units is the direct execution. This based on the fact that the computational semantics of both basic and composite units completely determined by their descriptions. This allows you to automatically generate a program in some programming language that implements the execution after compilation compiled assemblies. Compilation should be used for simple and basic constituent units, as their actualization requires large computational resources.

6. THE UPDATING AND RESUSCITATION OF THE UNIT

Actualization process is the change of it`s description, caused by changes in the environment. The characteristic feature of this approach is the ability to update created and run for the execution of processes in real time.

Indeed, the activation of the unit in a temporary storage area is copied to the full description, and the execution of the unit comes from the transfer of a description of some shell (Fig. 8). This organization allows the execution process after starting node to change it`s initial description. In this case, the unit will make the changes to the copy, and the execution of a new source of description.

During a major restructuring of host interface when there is a changing the number of actuators and events, as well as the number of input and output data, or data structures, the above method may not be applicable update. In this case, the update requires a redesign of the unit and of the composite units that use it as an internal one. Clearly, this is equivalent to the redesign and reengineering of the modeled process should be carried out by creating a new description.

However, at the stage of execution may be a certain number of copies of the updated site. In that case, the running unit in the updated state is in the process of resuscitation. Resuscitation unit is based on the knowledge of it`s current state and the states of all internal components, including the values of the input and output data, and by editing a copy of the description of the unit and its data according to the new understanding of the modeled process.

Thus, the proposed approach allows updating process models to meet changing or clarifying information about the process being modeled.

7. DYNAMIC PARALLELIZM

Notation of many famous modeling methodologies includes specific elements for an explicit (static) organization of parallel implementation. For this example, use tools such as logical AND, multipart cycles of parallel type, detailed processes with parallel units, gateway OR on a non-exclusive condition, etc. (Davis, 2005; Smith, Fingar, 2002). However, explicit control of parallel computation processes has significant drawbacks. First, qualitatively reduced visibility model, which is particularly important at the stage of analysis and verification. Secondly, parallel execution is lost much part of the processes run, parallelization of which may not be explicitly stated on the charts for many reasons. Thirdly, the development of a chart to parallel execution process is usually complex, highly qualified and unproductive labour.

Evolution of formalism combined control and data networks allows for the dynamic method (implicit) control parallelization based on the use of the natural parallelism of processes (Yatsutko, 2010b). Due to the fact that the constituent assembly contains an explicit reference to the received and transmitted data between internal nodes, when his performance parallelized to be executed all the activated units. Attention is drawn to the fact that it is here shown dynamic nature parallelization, which consists of the fact that a decision about which units are subject to parallel execution, and which are not is taken depending on the history of the modeled process (Yatsutko, 2012). It may be that at one and the same description of the unit at any one time shall be subject to the parallel execution of a subset of the internal units, and at another time – another one.

The most effective dynamic parallelizing manifested in cyclical interpretation of nodes, is when a unit selection for execution of the set is activated units, including belonging to different simulated processes. In this case we have a dynamic load balancing interpreters (CPU), which is manifested in the fact that, depending on the trajectory of the simulated processes more maintenance-process gets more computational resources.

Thus, the joint and several trace data and control between network units provides a flexible mechanism to synchronize the simulated processes, allows for natural parallelization and their effective implementation.

6. CONCLUSIONS

Application of the methodology for analysing and modeling processes based on the combined control and data networks can overcome existing problems in the field of industrial control business processes.

In particular, it is based on a graphical model of the process without the developer an opportunity to generate executable code. It is also proposed a graphical notation contains a small number of initial elements and simple rules of interpretation, is clear to analysts, as well as to other proven notation. The ability to update the models of processes running at their natural changes in the business environment, which can significantly improve the quality of management of modern dynamically changing enterprise, is also important.

The usage of the structured input and output processes creates new expressive means of facilitating verification posed models and the lack of modern modeling tools - namely, the interface specification input and output data, which can detect and correct errors during data tracing simulation.

Automatic dynamic parallelization process allows getting a high efficiency performance in a model. In this parallelism implemented is close to the natural parallelism of the simulated processes. Execution processes that use a dynamic parallelization different zooming into has a number of concurrent interpreters (processors, parallel threads). In addition, interpreters are automatically adjusted to run on the characteristics of the process execution, implementing dynamic balancing is loaded.

In the course of this research there has been created a prototype – a software tool which allows executing efficiently the models built according to the proposed approach.

REFERENCES

Beekman, D. (1989). CIMOSA: Computer Integrated Manufacturing – Open Systems Architecture. *International Journal of Computer-Integrated Manufacturing*, 2 (2), 94-105.

Booch, G., Rumbaugh, J., Jacobson, I. (2004). *The Unified Modeling Language Reference Manual, Second Edition*. p. 432, Addison-Wesley.

Cheremnykh, S. (2001). *Structural analysis of systems: IDEF-technology*, p. 208, Finances and Statistics, Moscow.

Davis, R. (2005). *Business Process Modeling with ARIS* A Practical Guide. New York: Springer-Verlag, p. 531.

Integration Definition For Function Modeling (IDEF0). (1993). p. 116, National Institute of Standards and Technology, Washington.

Fischer, L. (2002). *Workflow Handbook*, p. 428, Association with the Workflow Management Coalition (WfMC).

Gane, C., Sarson, T. (1979). *Structured Systems Analysis: Tools and Techniques*, p. 452, Prentice-Hall, New York.

Marca, D., McGowan, C. (1987). *SADT: Structured Analysis and Design Techniques*. p. 247, Mcgraw Hill Software Engineering Series, Mcgraw-Hill.

Mayer, R.J., Menzel, C.P., Painter, M.K. et al. (1995). *Information Integration for Concurrent Engineering (IICE). IDEF3 Process Description Capture Method Report. Ohio*, p. 235, Air Force Materiel Command.

Mertins, K., Rabe, M., Stiegennroth, H. (1993). Analysing production systems using petri nets based functional techniques, *Proceedings of the International Conference on Industrials Engineering and Production Management*, Vol. 2. Mons, Belgium: FUCAM, p. 961-970.

Poluha, R.G. (2007) Application of the SCOR Model in Supply Chain Management. Cambria Press.

Smith, H., Fingar, P. (2002). *Business Process Management (BPM),* p. 312, The Third Wave. Meghan-Kiffer Press.

Vykhovanets, V.S., Iosenkin, V.Ya. (2005) Concept analysis and context programming technology. Moscow, *Problemy Upravleniya*, p. 2-12.

Yatsutko, A. (2009a). Modeling the business processes of large-scale production. *Proceedings of the III International Conference « Managing the development of large-scale systems »*, Vol. II, p. 301-303, Institute of Control Sciences, Moscow.

Yatsutko, A. (2009b). Project Management based on a graphic description of the computational modeling of semantics. *Proceedings of the International scientific-practical conference «The theory of active systems»*, Vol. II, p. 61–64.

Yatsutko, A. (2010a). Updating of business process models in the combined control and data networks. *Proceedings of the Second Russian Conference with international participation « Technical and software control systems, control and measurement »*. Institute of Control Sciences, Moscow, p. 1268-1272.

Yatsutko, A. (2010b). Parallel computing methods in control theory based on the combined network and data management. *Reports of the 5th International Conference «Parallel Computations and Control Problems»*. Institute of Control Sciences, Moscow, p. 519-528.

Yatsutko, A. (2012). Synchronizing the data in an automation network with a separate trace control and data. *Proceedings of the Third Russian Conference with international participation «Technical and software control systems, control and measurement»*.Institute of Control Sciences, Moscow, p. 2060-2070.