Methods for monitoring of reconfigurable transport systems based on trigger functions

S A Nikishchenkov
Samara State University of Transport, 2, Svoboda Ave., Samara, 443066, Russia
E-mail: nikishchenkovs@mail.ru

Abstract. The article examines methods for monitoring of reconfigurable transport systems based on trigger functions. The methods of diagnostics are based on the information-logic scheme of the process.

1. Introduction
The development of transport systems (TS) features increased productivity, technical equipment and complexity in application of various economic strategies and adapting to changing operation conditions. Monitoring of TS correct functioning is carried out in order to minimize resources losses well-timed detection of defects, the causes of which include machinery malfunctions and failures, control and design errors, improper actions of personnel, abnormal situations, etc. [1]. The statistics and classification of defects based on sources, intensity, and consequences are specific for each TS, but on the railway transport there are tendencies in the growth of such factors such as rolling stock wear, shortage of skilled workers, power supply problems, quality of control systems, and conflicts of corporate and private interests.

The configuration is understood as the distribution of discrete operations of transport and production technology in a system with parallel operating and interacting functional units and in time [2-5]. Reconfiguration is the fact of the TS transition from one configuration to another, as well as structure and time-related redistribution of operations in their parallel flow, as caused by: asynchrony and pipelining of operations; system development or degradation; application of various control methods (algorithmization, dispatching, adaptation, self-organization); changes in economic, production, and technical strategies. Performance of a set of tasks in the reconfiguration conditions through the application of appropriate reconfigurability methods provides high performance, survivability, and availability indicators and is natural for a wide class of TS [5-7].

See Fig. 1 as an example of dependencies diagram between the basic operations a, b,…, r of the technology, being performed on N units (subsystems) of the TS, and the space-time diagrams of defect-free processes for configurations with N=4 and N=3, being present in the TS degradation.
Among the reconfigurable TS (RTS) studied by the Department of Informatics of the Samara State Transport University, there are: an onboard navigation fault-tolerant multiprocessor module, an analytical data collection system in the railway financial department, a motive-power depot, and network maintenance facility for freight cars.

According to the objects analysis, the known methods of functional diagnostics are not suitable to detect priority and specific defects, attributable to RTS, or these are ineffective, therefore the application of special approaches to solving problems of a diagnostic model selection, development of algorithms and means of defects detection, organization, and operation of monitoring system are useful [2].

2. Theoretical basis of the method

Parallel programming models and results are applicable not only to the PS productivity increase, but also to TS monitoring and diagnostics. This is due to the fact that paralleling is associated with a detailed study of processes information basis, with the TS monitoring being aimed on prevention of material and other resources loss. Trigger functions (TF) were proposed as a control mechanism for asynchronous calculations over a shared memory when the program statement was started under condition of the SF truth, preceding the statement body and depending, in general, on the execution of information and logical predecessors, as well as the process conditions and history [6-9]. For the RTS monitoring tasks, use of the TF supervisory properties rather than control properties is effective, while the monitoring tools are used to verify the parallel flow of operations according to TF value, with ensured invariance with respect to the structural and time distribution of operations as well as the process paralleling degree [10, 11].

Monitoring at the level of basic operations is advantageous because it features the defined composition, logical causation, and configuration of the process and provides sufficient rapidness in defects detection and a low level of resources loss.

The operation characteristics include the pertinence to the process, designation, type, location, input arguments and output results, function, time parameters, relations (connections) with other operations, and control parameters (procedure, order).

The theoretical and multiplex representation of the flow of operations, belonging to the same technology and distributed in the set of TS units in a monitored period of time includes:

- \( A_t \) is a subset of operations related to the t-th technology.
- \( A_t^r \) is a subset of operations being started in the moment t.
- \( A_{i,t,n} \) is the i-th operation triggered in the moment t in the n-th functional unit.
- \( A_{i,t,n}(P_j) \) is the operation for logical-condition check.

![Figure 1. Example of TS reconfiguration](image-url)
- \( inpA_i \) and \( outA_i \) are input and output operations.
- \( fA_i \) and \( tA_i \) are the function (conversion) of operation and its running duration.
- \( A'_i \) is a subset of operations being “ready” for launch under conditions of all resources availability.

The method is based on the analysis of ratios in the process operations flow using process parallel models in the form of statement schemes, TF, data and logic schemes, and counter networks [9,12]. Knowledge of the monitoring object, resulted from such object formal representations and studies of its sets of correct realizations, are used for RTS defects description and detection.

The statement scheme in Fig. 2a describes the fragment of the repair process of a diesel-locomotive unit and represents the set of: operators \( A_1-A_{15} \) (\( A_8 \) and \( A_9 \) are to check the value of \( P_1 \) and \( P_2 \) logical conditions), n-tuples of input and output variables (\( inA_6 = a, b; outA_6 = f \)), transfer of variables between statements and their run logic (for cyclic repetition and alternative execution). The interpretation of the circuit is as follows: \( A_1-A_5 \) are corresponding to the input acceptance of parts and components \( a, b, c, d, e \); \( A_6, A_9, A_{12}, A_{14} \) are the cyclic processing of the \( f \) assembly with measurement \( P_2 \) and transfer of \( f \) to the final assembly of the unit \( A_{13} \); \( A_8 \) checks \( e \) for \( P_1 \) tolerance and provides an alternative transmission to \( A_{13} \); \( A_{15} \) completes the assembly.

Statement scheme, conventionally used to describe programs, is a comprehensive tool for the formal representation of basic and auxiliary transport processes (transportation, loading-unloading, repair, maintenance, etc.).

TF is an ultimate predicate over multiple statements and logical conditions in the scheme (including, in general, additional ones, which consider the process history) and specifies the maximally parallel asynchronous process (for the class of so-called “free” schemes), to ensure statements start as far as their input arguments are ready. The graph in Fig. 2b describes such control with two types of logic used: AND and Exclusive OR (\( \& \) symbols are accepted by default and not shown, the Exclusive OR is denoted with \( \oplus \)). For example, for the \( A_{13} \) statement, the start condition will be as follows:

\[
C_{13} = A_6 (P_2^1) \& A_{10} \& (A_8 (P_1^1) \oplus A_{11}) \rightarrow A_{13}
\]

i.e., the \( A_{13} \) will be started in case of \( A_6 \) was executed with \( P_2 = 1 \) value and \( A_{10} \) was executed, and either \( A_8 \) with a value of \( P_1 = 1 \) or \( A_{11} \), is executed. The TF-based control assumes the existence of an appropriate architecture that provides the operations flow with the fulfillment of relation:

\[
\forall t A'_i = A'_i = \{ A_i.t(C_{i,t} = 1) \}
\]

The properties of TF as a result of the scheme maximum paralleling allows the effective use for process monitoring purposes. In this case, based on the TF value, the correctness or defect of the statement start during the controlled period of time is checked. For \( A_{13} \) the monitoring function is:
\[ K_{i,j} = A_{i,j} \leftarrow A_0 (P^i_j) \land A_{10} \land (A_8 (P^i_j) \oplus A_{11}), \]  
and the start is considered correct, provided the \( A_0 \) was executed with \( P_2 = 1 \) value and \( A_{10} \) was executed, and either \( A_8 \) with a value of \( P_1 = 1 \) or \( A_{11} \), is executed.

For correct (defect-free) processes, several assertions are valid.

1. Any process may be represented in the statement scheme form, and for each statement a trigger and monitoring function can be set.

2. For process with internal parallelism, there are many ways of \( A^i \) (control methods) and correct equivalent workflows generation, with different configurations:

\[ \forall \ i. A^i \subseteq \left\{ A_{i,1} | K_{i,1} = 1 \right\}. \]

3. In the workflow of the correct process, following relation applies:

\[ \forall \ i. \forall A_{i,1} \subseteq A^i \subseteq A^i \subseteq A^i. \]

4. For each process there is a correct maximally parallel asynchronous pipelined workflow corresponding to (1).

5. Any configuration of the workflow is an additional structure and time-related ordering of the maximally parallel workflow.

The aggregate of control functions for all (or selected) scheme statements, given in the form of string expressions of type (2) or a biology graph, is called the process data logical scheme (PDLS) [10,11,16].

Within the PDLS algebraic formulation there is a triplet \( S=(A,L,F) \), where the \( A \) set consists of transducer and recognizer statements, a set of predicates \( L=PUQ \) includes the \( P \) process logical conditions and additional predicates \( Q \) describing its history, and the aggregate \( K=\{K_i(A,L)\} \) represents a set of logical functions in a Zhegalkin algebra basis (AND, Exclusive OR, 1), which means the running of predecessors from \( A \) for the statement \( A_i \) for set values of \( L \).

The PDLS graph will be represented as \( G=(A,B,L,\&,\oplus) \), where \( A \) is a set of peaks; \( B \) is a set of curves, interpreting relations of logical and data precedence between statements, such, where \( B_{i,k} = 1 \) for peaks \( A_i \) and \( A_k \), if \( A_i \) is included as an argument to \( A_k \); each peak \( A_i \) is marked with input logic using \( \& \) links between input curves; each peak, corresponding to recognizer statement \( A_i(P_j) \), is marked with output logic using \( \oplus \) link between alternative output curves, loaded with various values of \( P_j(P^1_j \) or \( P^0_j \)), and \( \& \) link between the curves, loaded with identical values of \( P_j \); without transitive closures.

The PDLS describes required data and logic relations of statements precedence as well as conditions for their correct asynchronous (including pipelined) start, i.e. is an invariant for a set of equivalent processes, differing in the structure-time distribution per TS units and control method applied [10]. According to [12], the PDLS in the general case is a simple, nonplanar loosely coupled, correct control graph with special marking of peaks and curves, the semantics of which is designed for systems flow analysis. The PDLS dynamics may be described as a process, consisting of such events as peaks and curves “inclusion,” while routinely simulating pipeline processes [10-14]. The PDLS synthesis and analysis problems are considered in [10], while the presence of a wide range of de-sequence programs ensuring the automation of parallel algorithms formation based on a given sequential scheme is significant [15].

The PDLS correct run features the set of started statements; \( A^i \) is a non-empty subset of “ready” statements from \( A^i \):

\[ (A^i \subseteq A^i) \land \left( |A^i| > 1 \right). \]  

Exceptions to the rules are represented with defects as follows:

\[ (\exists r \exists A_{i,r} \not\subseteq A^i) \lor \left( |A^i| = 0 \right), \]

i.e., two types of defects are defined for PDLS: “false start of statement” and “no start.” As it was proved in [10], the “statement replacement” defect with the TF-based control is reduced to the “false start” defect.

The aggregate of the specific PDLS and relations (3) and (4) represents the RTS diagnostic model. As
compared with algorithmic control, this method with the same detection ability (on the class of defects, reduced to the “false start”, “replacement”, “no start” list) allows one to control the reconfigurable processes (sequential, statically and dynamically parallel, pipelined, with arbitrary dispatching and control method, self-organizing) with the same tools settings, i.e. PDLS is a diagnostic invariant of RTS [10,16]. The model features versatility, interpretability and adequacy, the formal issues resolved, compactness, decomposition ability and operability [1-3,10].

3. Method implementation principles and means
The RTS monitoring principles include [15]:
1) the RTS diagnostic model is represented as the PDLS on the level of technology’s basic operations;
2) the PDLS is interpreted by the counter network for defect detection algorithms and tools implementation;
3) in the RTS operation, the events are recorded in the workflow, the diagnostic indicators are processed and defect indicators are issued by monitoring aids;
4) in order to improve the monitoring efficiency, the graph decomposition of PDLS and marking with process additional parameters are used.

The use of monitoring counter networks (MCN) is a practical way of transition from a graph diagnostic model to the creation of hardware, software or organizational and technical monitoring tools. Each PDLS element is associated with a network element, and each event occurring in the monitoring object at the operation level is reflected in the network, which recognizes its correctness or invalidity, depending on its state and the process flow. The network element consists of a counter and logic circuits, generating an event-related acknowledgment or defect signal. The counter interpretation of the TF is natural, since the elementary TF is a trigger. The MCN implementation is quite simple at the hardware (homogeneous environments, special processors) and at the software level (modules within the ACS, diagnostic processors in spreadsheets, DBMS). Due to the fact that the MCN “skeleton” is PDLS, processes with a different (including maximum) degree of parallelism can be monitored. The network is set up for a specific process monitoring by switching (addressing) the elements according to a given statement scheme.

In [11], a minimal MCN is proposed, featuring same number of one-bit elements as the number of monitored operations (15 for the process in Figure 2). The statement start is confirmed provided its predecessors are executed and the element has passed from the initial state 0 to 1; otherwise a defect signal is generated. This type of MCN monitors processes with arbitrary statement scheme and paralleling degree, but the class of defects is limited, since the operation duration and phases are not considered.

In order to solve the dimensionality problem, the procedure of decomposition into parts and isomorphic fragments, vertical (trajectory) and horizontal (tiered) decomposition and homomorphic contraction may be performed with the PDLS graph [11].

See the Figure 3 with diagrams of MCN options with extended capabilities, where α, and αc are attributes of operation start and completion, coming to the network from the monitored object; αp and αc are attributes of confirmation of correctness of operation start and completion, generated by network; f, fV, and f are logic functions, describing conditions of preceding operations execution; d is a defect attribute in a process related with this operation.

The network with elements, having 3 states (Fig.3, a), detects the false start or completion of operation in condition 0 (“not ready”), goes to state 1 (“ready, waiting for start”) based on actual value of availability function f, and in this state it detects the false start defect (not all predecessors have run), goes to state 2 (“start, begin of run”) with the operation start signal from the object (provided the admissibility function f is true), returns to state 0 on a signal of operation completion. Functions f, fV, f
and $f^*$ are derived from the decomposition of the PDLS statements monitoring function [11].

The MCN with monitoring of PDLS curves and peaks contains two types of one-bit elements, see

![Diagram A](image1)

and

![Diagram B](image2)

**Figure 3.** Diagrams of MCN options

Figure 3, b. Run of $a_n^{pc}$ statement includes curves, coming from $A_n$ (according to output logic), monitoring function $f_i$, is interpreted by the state of curves elements for peak $A_i$ (according to input logic), the element of peak (of operation) is transferred from the state 0 to 1 (“running”) based on operation start and provided the function is true. The number of MCN elements for the PDLS monitoring in Fig. 1 is 31, but due to an additional “layer” of curve elements, it features increased detection capabilities, for pipelined processes as well.

**Conclusion**

The method advantages are determined by the PDLS diagnostic properties, the model versatility, elaboration of theoretical issues, as well as availability of paralleling programs. The main problems of practical application include expert evaluation of process descriptions, the RTS controllability, complexity of engineering tasks (interpretation, visualization, simulation of RTS correct and defect operation in conjunction with monitoring means, availability of support tools), organizational and technical implementation, methods of defect situations response, reduction of losses and ensured operability [2,15].

**References**

[1] Czichos H 2013 *Handbook of Technical Diagnostics. Fundamentals and Application to Structures and Systems* (Berlin: Springer-Verlag Berlin Heidelberg)

[2] Ignatushchenkov V V 1984 *Organization of control structures of multiprocessor computer systems* (Moscow: Energoatomizdat)

[3] Narinyani A S 1974 The parallel programming theory: formal models *Cybernetics* 3 4

[4] Kotov E V, Narinyani A S 1966 Asynchronous computing processes over the memory *Cybernetics* 3

[5] Kotov E V 1978 *Introduction to the theory of program schemes* (Novosibirsk: Nauka)

[6] Padua D 2011 *Encyclopedia of Parallel Computing* (New York: Springer)

[7] Nikishchenkov S A 1988 *Functional diagnostics of control part of reconfigurable multiprocessor computer systems with information schemes of algorithms*. (Leningrad: LETI Technical Universitet, Phd)

[8] Kasyanov V N, Evstigneev V A 2000 *Graph Theory for Programmers. Algorithms for Processing Trees* (Boston: Kluwer Academic)
[9] Ryzhkov A P 1976 Correct bilological graph simulation of parallel computational process and its properties // Notice of USSR Academy of Sciences. Technical Cybernetics 2
[10] Golovkin B A 1983 Calculation of characteristics and planning of parallel computing processes (Moscow: Radio i svyaz)
[11] Bunich A L et al 2002 Parallel calculations and control tasks (analytical review) Automation and Remote control 12
[12] Nikishchenkov S A 2001 Principles of functional diagnostics of control systems based on information and logic schemes of processes (Moscow: VNTIC, No. 73200100204)
[13] Shcherbakov N S, Podkopaev B P 1982 Structural theory of digital automation monitoring (Moscow: Mashinostroenie)
[14] Yusipov R A 2003 Self-learning of consideration model of weather conditions impact on process operations duration. Railway Research Institute Bulletin 5
[15] Nikishchenkov S A, Priputnikov A P, Mikhaylov N V 2001 Software and organizational and technical monitoring tools of conveyor systems Proceeding XI Russian Scientific Conference PSATI (Samara: PSATI) pp. 174-182