engineering proceedings

MDPI

Proceeding Paper

Application of a Deterministic Optical Network Model for the Implementation of an Expert System Knowledge Base for Information Transmission Failure Management †

Vladislav Kosyanchuk [1], Nikolay Selvesyuk [1], Evgeniy Zybin [1], Valeriy Novikov [1,*], Valentin Olenev [2], Andrey Solovyov [3] and Mikhail Semyonov [3]

[1] State Research Institute of Aircraft Systems (GosNIIAS), 7 Victorenko St., 125167 Moscow, Russia; vvk@gosniias.ru (V.K.); nis@gosniias.ru (N.S.); eyzybin@2100.gosniias.ru (E.Z.)
[2] Department of Aerospace Computer and Program Systems, Saint-Petersburg State University of Aerospace Instrumentation (SUAI), 67 Bolshaya Morskaya St., 197000 Saint-Petersburg, Russia; valentin.olenev@guap.ru
[3] Department of Digital Technologies, Voronezh State University (VSU), 1 Universitetskaya pl., 394018 Voronezh, Russia; darkzite@yandex.ru (A.S.); mkl150@mail.ru (M.S.)
* Correspondence: vmnovikov@2100.gosniias.ru; Tel.: +7-499-157-70-47

† Presented at the 15th International Conference "Intelligent Systems" (INTELS'22), Moscow, Russia, 14–16 December 2022.

Abstract: This article presents an approach for the construction of an expert system, which helps to make the decisions when parrying failures during information exchange in real-time systems based on a deterministic optical network. The authors analyze the requirements for deterministic data exchange environments and consider the trends for the development of software and hardware for deterministic networks. The main goal of this article is to propose a concept for the implementation of a deterministic control unit based on an all-optical network with spectral division multiplexing. Two main steps for the implementation of such a system are described. The first is the implementation of the DOS protocol stack and its model, which was used in several protocol development stages. This model is a part of a proposed concept for an automation design and development system for information exchanges. The second presented step is the development of an expert decision support system, based on the principles of the supervisory approach. This article presents a neurocontroller, which provides hardware support for the implementation of individual modes of operation of an expert system.

Keywords: distributed information computer network architecture; deterministic optical network; network model; dynamic reconfiguration; design automation system; expert system; decision support system; neurocontroller; WDM network; system on a chip

check for updates

Citation: Kosyanchuk, V.; Selvesyuk, N.; Zybin, E.; Novikov, V.; Olenev, V.; Solovyov, A.; Semyonov, M. Application of a Deterministic Optical Network Model for the Implementation of an Expert System Knowledge Base for Information Transmission Failure Management. *Eng. Proc.* **2023**, *33*, 51. https://doi.org/10.3390/engproc2023033051

Academic Editors: Askhat Diveev, Ivan Zelinka, Arutun Avetisyan and Alexander Ilin

Published: 19 July 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The main goal of this article is to consider the development and operation of a deterministic control unit (DCU) for an arbitrary technological process in real time (RT). Modern onboard equipment and networks increase in their complexity from year to year, with the number of tasks undertook by such systems growing significantly. Therefore, the industry needs to ensure a high level of reliability during the operation of such systems. For this purpose, aircraft onboard networks' developers need effective mechanisms for ensuring deterministic data delivery. DCU is a deterministic real-time technological complex, which controls a technological process (TP). Its operation depends on external influencing factors, which move DCU from an internal state to a uniquely defined (deterministic) new state in a specific predetermined time interval. DCU is based on a distributed information computing environment (RIVS) [1] and built of the computing modules (CM) of the same type. A network should connect these models. There are several elements, which are the typical

RIVS modules: N computing modules, K control loops for technological systems, L data concentrators. All of these modules have access to a local information system (LIS). Most TP problems are solved using RIVS, but the time for solving the task should not exceed a certain maximum task duration time. The time for each task from the control loop is determined from the permitted operating time of a separate system control loop and the entire DCU. The key DCU functioning process is the exchange of the information between its subsystems. Transmitted or received data determine the correct operation of functional software applications (FSP) and the whole TP.

2. Local Information System Requirements

Several DCU LIS features distinguish it from other real-time and information networks:

- A limited number of CMs (50–100);
- The total amount of FSPs, which, for functionality, does not exceed 256 programs;
- All information flows operate in strong correlation with specific communication protocols;
- The network provides the transmission of control signals in the hard RT mode [2–4].

Such control signals are time markers used for the synchronization of the system, and control and notification signals [5–10].

Typical LISs face problems of data transmission delays, which makes the operation of the system unpredictable and leads to packet collisions. Of course, it is possible to significantly increase the data speed or use specialized type of quality of service (QoS). This QoS is scheduling, which ensures data transmission in accordance with a schedule specified for the whole network during the configuration phase. However, unfortunately, during the operation of the DCU, unexpected delays and collisions occur. This produces an effect on the work of the scheduler, so existing networks cannot provide complete determinism, which is very important for onboard networks. Regarding this, the main sources of collisions should be excluded from the network architecture to ensure complete determinism. Most such delays are caused by switches and routing mechanisms.

The construction of LIS DCU should be based on the following principles: the "point-to-point" organization of the data exchange between the FSP of different CMs; using broadcast mode for sending packets as the main mode; using of distributed memory for data exchange between FSPs. Such LIS principles could be implemented in deterministic real-time optical networks (DOS RV). Such networks transmit the packets without switching and use distributed memory principles for data exchange between end systems. All-optical networks are a type of networks where the switching, multiplexing, and data transmission are organized by purely optical technologies, rather than by electronic technologies. DOS RVs are built as an information transmission system with two types of topologies: a star and a ring. They contain N VMs with the FSPs function and a spectral SDA (SSDA). Figure 1 shows a block diagram of the DOS RV.

Some key principles should be taken into account to build an effective DOS RV [11,12]. The optical technology of wavelength division multiplexing (WDM) should be used as a transport architecture. WDM consists in the transmission of many different data flows over the single optical fiber, but in separate optical λ-channels. Each channel has its own optical frequency called the λ-channel. Information exchange is based on the principle of distributed shared memory (DSM). Memory access is performed via the simple "read" and "write" commands. Each FSP has a particular memory area, which is used to exchange data with external subscribers of the fiber optic medium. It is performed via the λ-channel specially allocated for this FSP, and this frequency is fixed for the data flow in terms of the whole network.

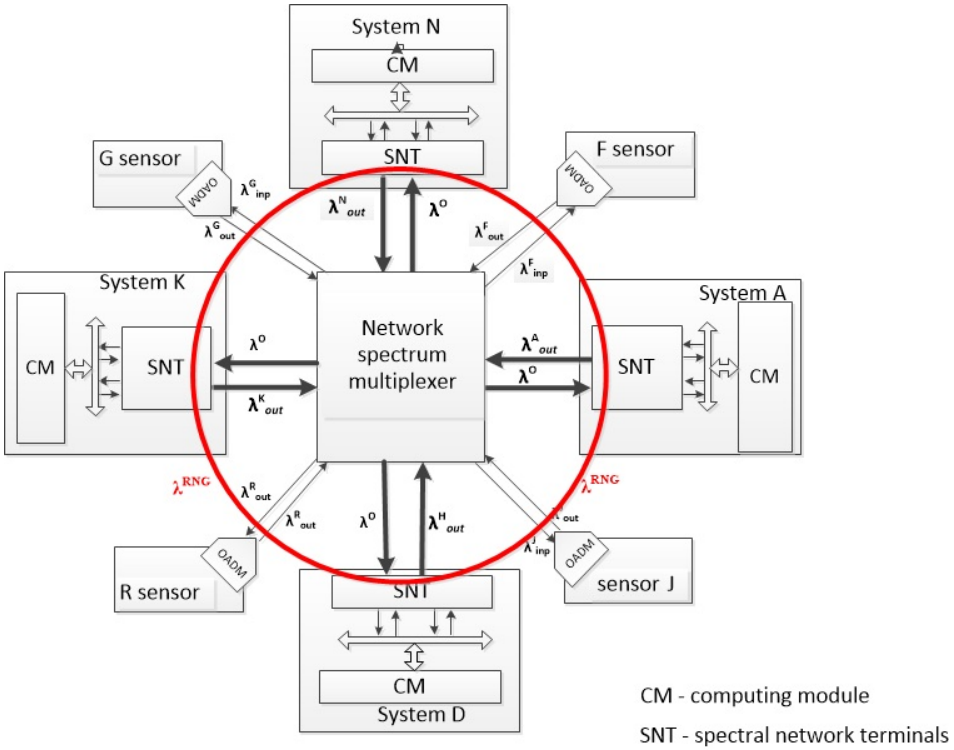


Figure 1. DOS RV structure.

There are a few important steps to organize the work of DOS RV:

- Form a list of RIVS systems, which participate in the data exchange of the real-time information transmission system—N;
- Form a private list of FSPs operating in each of the VM of RIVS modules. These modules could be logically split into two groups. The first group includes the total number of FSPs transmitting information L_{out}. The second group includes the total number of FSPs receiving information l_{inp}. Both groups form a general list of FSPs transmitting information throughout the RIVS—L_{out};
- Set the number of channels that are required for all information exchange between FSPs. $L_{out} \leq \lambda^0$ must be true, where λ^0 is the total number of spectrally multiplexed λ-channels in DOS RV;
- For each system, determine the total number of output λ–channels. For example, for system A, the group of channels is defined as $\lambda^A_{out} = U\{\lambda_1 \ldots \lambda_{nA}\}$. These groups have their own set of wavelengths for a given SSDA;
- In the same way, determine the total number of input l_{inp} in the SSDA of each RIVS component.

3. The DOS Model in the Automation Design and Development System for Information Exchanges of Real-Time Systems

To design communication protocols that implement information exchange in the DOS RV networks, modern methods should be used. Such a methods should cover all the system development stages (see Figure 2).

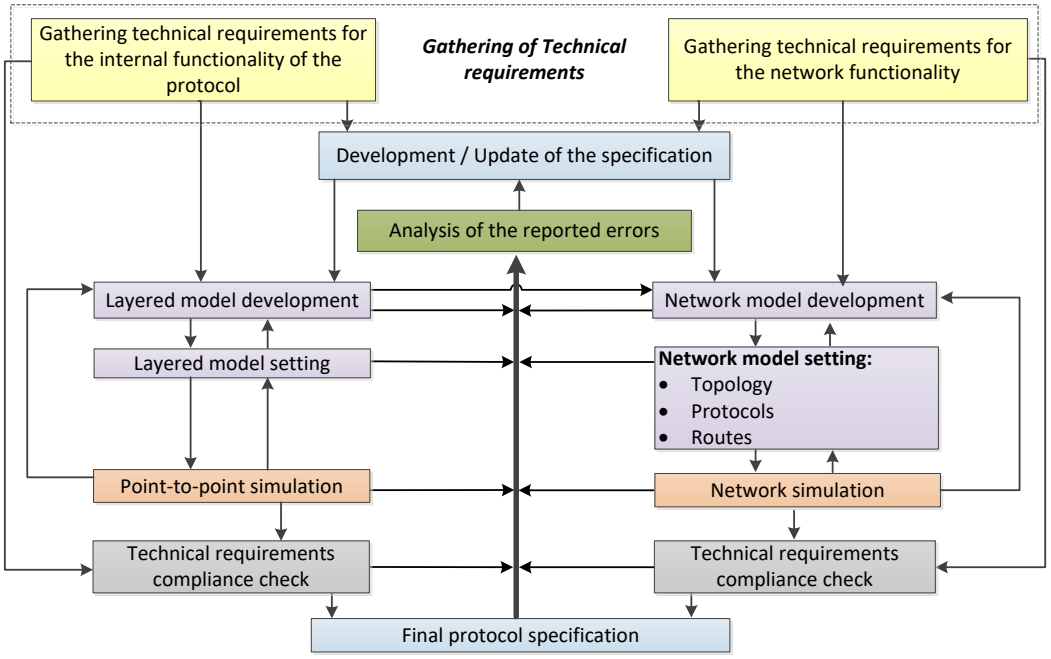


Figure 2. Methodology for development of communication protocols for real-time systems.

The design process begins with the gathering of technical requirements for the future protocol and their analysis. Technical requirements are divided into requirements for the internal mechanisms of the protocol and requirements for the operation of devices based on this protocol. Regarding this, the first version of the specification of the future protocol was developed. A simulation was used to test and verify the developed protocol.

The modeling process creates layered and network models. The network model is created after at least one successful version of the layered model has been created. Each of these types of models solves its own problems and checks the operation of different mechanisms of the protocol. After creating the models, it is necessary to configure them, and then carry out detailed modeling. If an error is found at any of the parallel stages, it is necessary to create a bug report, send it to the expert group for review, and stop work in the parallel branch until the specification is checked or clarified. After analyzing the error, the specification is refined and remodeled. When both simulation teams report that no more errors have been found, the simulation results are compared to the original technical requirements. If the requirements are not met, the specification is sent back for revision, otherwise the document is finalized and can be sent to further stages of the project, for example, hardware implementation.

The gathering of technical requirements is carried out by an expert group, which includes representative customers, developers, researchers who have the necessary knowledge and information on the further use of the protocol. The completeness of the specifications is an important issue, as the requirements will be the basis for deciding whether there are errors in the protocol.

Next, the specification of the designed communication protocol is written. Such a document usually includes a general description of the protocol stack (or a single protocol), data transfer formats, protocol mechanisms and algorithms in accordance with the layered architecture, interlayer interfaces, and examples of the application of the above mechanisms or the protocol as a whole. Once this task is completed and the initial draft of the specification is ready, the document is submitted for modeling. Modeling is carried out in order to check the underlying algorithms and can be carried out already in the process of writing the specification. To do this, an intermediate document is already given for

modeling; it also passes through various checks, but is returned back to the revision stage if such a specification does not meet the technical requirements.

Further modeling should be carried in two parallel branches: layered protocol modeling (light gray blocks in Figure 2) and the system of devices modeling (dark gray blocks in Figure 2). When modeling a layered protocol stack, the model layers fully correspond to the protocol structure, and the interfaces between these layers correspond to service access points. Thus, to carry out the simulation, it is sufficient to fully describe the operation of the protocol in relation to one device. To obtain a result, it is necessary to organize data transfer between two devices only, since, for the task of analyzing the protocol, network interaction and the mechanisms of operation as part of a larger system do not matter. When writing a device interaction model, the network operation and the exchange of various information packets are modeled. With this approach, the interaction of the constituent elements of the processes inside the device is not considered. In order to effectively use the communication protocol modeling, both of these approaches should be used. This is the only way to detect most of the errors in the specification.

The development of a layered model can be carried out with software and formal models can be created using well-known mathematical tools. For example, the Petri nets theory, the automata theory, queuing systems, etc. can be used. Programming languages based on formal methods could be used, for example, SDL, GPSS, etc. Setting up such a model consists of setting protocol parameters, and other properties that should be described in the specification.

For modeling a system of devices, programming languages are chosen that can build large systems with the use of multiple modules, ensuring the parallel operation of these modules. One of the most efficient languages for these purposes is SystemC. It covers a wide range of protocol development areas and can be used for architecture design, hardware and software simulation, system behavior description, and functional verification. Finally, SystemC supports hardware modeling and design. Using the SystemC network model, you can set various configuration parameters, such as the data rate, number of nodes and switches, time delays, routing tables, number of ports in switches, etc. Also, on network models, it is possible to automate the process of designing a network topology, the tracking of routes, and setting various network properties.

During the tuning of the model, some errors in the model itself can be detected and, as a result, a request is sent to correct the model. During the model development phase and the modeling phase, errors in the designed specification can be found. In this case, a detailed description of the error is generated. The team developing another model in a parallel methodology branch is notified that the modeling process should be temporarily suspended. Next, an expert evaluates what changes are needed in the specification, and then corrects it. When changes are made to the models, the specification is resubmitted for modeling.

Once the simulation process is completed successfully and no more specification errors are found, the simulation results are compared to the original specifications. In case of discrepancy, the specification is sent for correction. Otherwise, the project team creates the final version of the specification and submits it for implementation [13].

Since modeling is such an important part of the communication protocol development process, it is necessary to have an automated system that allows one to set up a software model and run it for execution. Such CAD systems are becoming an integral part of the technology, since they can significantly speed up the development and reduce the cost of the process of creating new technological solutions. Let us consider what the process of creating such a CAD system for DOS RV networks could look like.

Modern onboard networks consist of a large number of elements with different functionality. All these elements are interconnected through the infrastructure of the onboard network. Such networks are easy to operate as long as they are small enough, and as long as they have predictable behavior and are used for a simple set of information flows without strict requirements and restrictions. However, as the network grows larger, infor-

mation flows become denser, requirements and constraints become more stringent, and design becomes a challenge. In such cases, there is a need to create a software packages for computer-aided design and network modeling.

For DOS RV networks, there is a need to create an automated design and simulation system that will cover the full cycle of development and simulation of the onboard network, starting with the design of the network topology and ending with obtaining network simulation results and statistical data (see Figure 3).

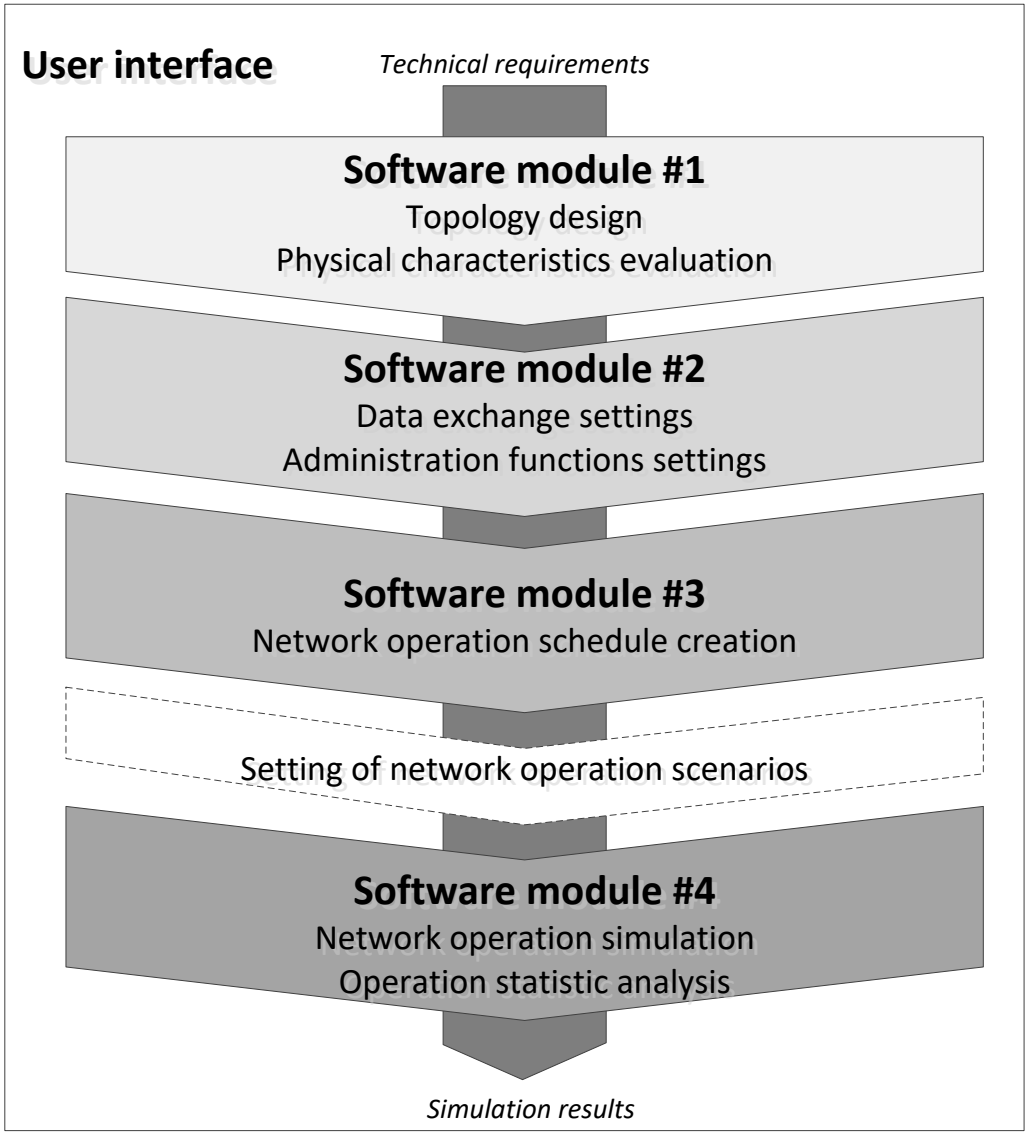


Figure 3. Architecture of CAD for DOS RV.

The designed CAD DOS RV will use a graphical interface that allows for an optical onboard network to be assembled from a variety of network nodes and a spectral multiplexer (in the future, several multiplexers) corresponding to the physical implementations that are available for building networks. A configuration interface will be provided for setting parameters, configuring protocols, and test scenarios.

Software modules (SM) will perform functions corresponding to the various stages of the onboard network design. SM-1 is responsible for designing the network topology and the evaluation of the physical characteristics, such as the mass of cables and devices, power consumption, and the evaluation of the required number of data exchange channels in the network. SM-2 is responsible for complete network configuration, including functions for configuring administration protocols. SM-3 is used to adjust the "Scheduling" quality of

service implemented in the DOS RV protocols. At SM-3, the design work is completed, and at SM-4, the entire network is launched for simulation. The simulation proceeds in accordance with the specified work scenarios, which imply turning off devices, introducing errors into the network, changing settings, etc. Upon completion of the simulation, the user has a complete set of diagnostic information on the operation of the DOS RV and graphical information on statistics.

To implement the simulation functions, it is necessary to create an effective simulator of the DOS RV operation. This model can be used to solve the problem of modeling protocol stack specifications, testing mechanisms, and testing prototypes of future real networks on the model. The developed model allows one to work with the simulation environment in a graphical user interface, perform simulations of various scenarios, and generate detailed diagnostic information. As the first step in the implementation of the DOS RV protocols, a point-to-point network model was created. It consists of two nodes and a data transmission channel. Using the developed model, a detailed study of the mechanisms of the transport layer, the link layer, and the administration layer was carried out and the efficiency of the developed mechanisms was shown. To do this, we tested the transmission of various packets with different qualities of service, interrupts, time-codes, commands for working with distributed memory, and requests for the administration level [14].

This model will form the basis of the extended version, which will be able to set flexible work scenarios and use the full DOS RV network infrastructure. The model is implemented in the SystemC language, which is used to model parallel systems. In the future, we plan to use this model as part of an expert system for preparing and making decisions in the event of an exchange failure in DOS RV.

4. The Concept of Applying the DON Model to Form the Knowledge Base of the Expert System for Preparation and Decision-Making

Currently, an urgent problem is that of designing real-time onboard intelligent systems. Such systems are based on the integration of models capable of adapting, modifying, learning, representing, and operating knowledge, focused on the specifics of the problem area and the corresponding type of uncertainty, which reflects their ability to develop and change their state.

When implementing a real-time decision support and decision-making (RTDSDM), it is necessary to take into account the specifics of such systems. The system should obtain a solution under the time constraints determined by the real controlled process. It should take into account the time factor when describing a problem situation and in the process of finding a solution. Also, it is impossible to obtain all the objective information necessary for the decision, and, in connection with this, the use of subjective, expert information. There is a need to have a multivariate search, and the need to apply methods of plausible inference and active participation in the process of the search and intellectual inference. Case-based decision search methods can be used in many RTDSDM blocks (analyzer, decision search blocks, explanations, modeling, and forecasting) and can improve the efficiency of decision-making in various problem situations.

Precedent-based intelligent inference is an approach that allows a new problem to be solved by using or adapting a solution to an already known problem, i.e., using the already accumulated experience in solving similar problems.

The disadvantages of case-based reasoning include the following:

- When describing precedents, they are usually limited to superficial knowledge about the subject area;
- A large number of cases can lead to a decrease in system performance;
- It is problematic to define criteria for indexing and comparing precedents;
- Difficulties with debugging algorithms for determining similar (similar) precedents;
- The impossibility of obtaining a solution to problems for which there are no precedents or the degree of their similarity (similarity) is less than a given threshold value.

The main purpose of using the apparatus of precedents within the framework of RTDSDM is to issue a ready-made solution to the operator for the current situation based on precedents that have already taken place in the past when managing a given or similar object (system).

It is obvious that in order to structure information and quickly access it from the RTDSDM core, it is necessary to form a specialized knowledge base (KB) containing a set of precedents, each of which describes an emergency situation and certain methods for solving it. One such solution, which allows for parrying the failures that have arisen, is the dynamic reconfiguration of the DOS RV. In addition, information about an emergency situation can be taken both from the real experience of a qualified pilot, and formed using simulation modeling implemented in CAD based on the DOS RV model. The DOS RV model can also be used in RTDSDM to analyze the current state of the network and predict failure situations. This function is implemented by the method of the continuous comparison of the functioning of the network and the DOS RV model implemented in the core of the RTDSDM.

The intelligent inference function is produced by a specialized expert system (SES), which is part of the RTDSDM. To increase flexibility and expand the ability to solve a wider range of problems, SES should rely on both classical search algorithms using standard calculators and specialized parallel algorithms implemented on the basis of coprocessors. One of these coprocessors is a neural network computer or a neurocontroller (NC). On the basis of an NC, it is possible to deploy artificial neural networks with different architectures and different specifics of functioning, which will allow hidden connections and nonlinear patterns to be identified in the data stored in the knowledge base, which, in turn, will significantly increase the flexibility and depth of intellectual inference carried out in the SES. The generalized structure of the RTDSDM is shown in Figure 4a.

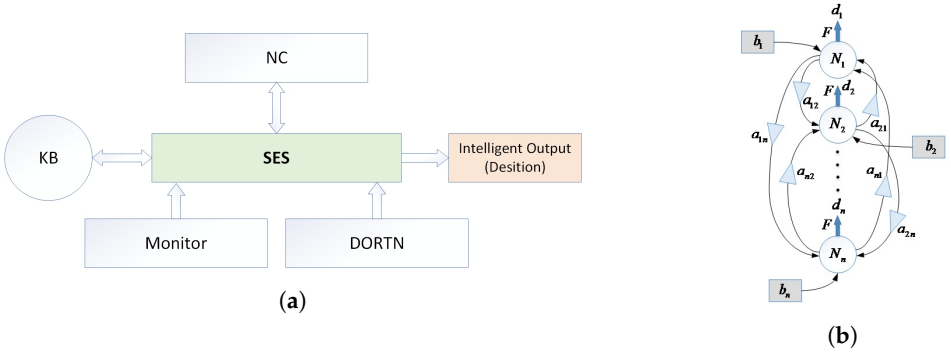


Figure 4. (**a**) The structure of the RTDSDM based on the DOS RV model; (**b**) NC architecture.

As can be seen from Figure 4a, the RTDSDM includes a tera-network calculator or neurocontroller. This element can be realized using a microprocessor, on the basis of FPGA, and in the form of a special module representing a system-on-chip (SoC). Depending on the variant of realization, the pro-cems in the NC can be sequential (in the case of a microprocessor), parallel (in the case of an FPGA), and combined (in the case of a SoC). The NC architecture is shown in Figure 4b.

As can be seen from Figure 4b, a NC consists of neurons N_j, whose inputs are multiplied by weight coefficients a_ij; each neuron transforms the weighted sum of inputs using the activation function F_j and the output is formed d_j, where $i = j = \overline{1,n}$. Taking into account the peculiarity of neuron architecture, namely, the presence of many inputs and only one output, the NC can be described using the neuron interaction matrix M^{nxn} (NIM), whose elements outside the diagonal are the weights of the inputs of the neuron with the number corresponding to the row number, and the diagonal element is the output of this neuron.

In addition to the NIM, it is also necessary to describe the set of activation functions used in the NC with reference to each neuron. For this purpose, we introduce the vector of activation functions F^n (VAF), whose elements f_i indicate the number of the activation

function implemented in the neuron N_i. Odds a_{ij} in the NIM affect the slope of the activation function of each neuron; however, to increase the flexibility of the NC, it is also necessary to introduce bias coefficients b_i, which allow one to move the activation function of each neuron to the left or right. To this end, we introduce the vector of bias coefficients B^n, containing b_i. Thus, the NC is mathematically uniquely described by the set

$$\{M^{nxn}, B^n, F^n\}, \quad (1)$$

where n is the number of neurons in the NC. The NC dynamics can be described by the following transformation:

$$d_t = (M \times d_{t-1} + b) \quad (2)$$

Due to the fact that the NIM is used as part of the SES, it is possible to significantly increase the flexibility of decision-making and increase the speed of some (e.g., nonformalizable) computations. This is possible due to the fact that on the NIM, it is possible to implement a large number of different classes of neural networks (multilayer perceptrons, Hopfield network, Boltzmann machine, etc.), both direct propagation and recurrent, without changing the NC architecture. It should also be noted that, in the case of using a parallel implementation of NIM (on FPGA), the response time of the NC is determined by the FPGA production technology and currently reaches units of nanoseconds.

5. Conclusions

The use of a commutation-free optical environment in onboard systems and the construction of a distributed system of support and decision-making on its basis allows one to significantly increase the flexibility and scalability of the onboard equipment complex, simplify and accelerate the process of its reconfiguration (for example, in the case of dynamic reconfiguration), and to achieve high determinism of states of all onboard processes and systems. It should also be noted that the use of the information network model, which is formed and used in the CAD environment during the development of DOS RV, allows one to obtain a dual-purpose model (both at the design stage and at the stage of intelligent control of the information network environment), to create a single, continuous process of development, and the application of DOS RV. In general, the efficiency of work is significantly increased and financial costs are optimized. The considered approach is methodically applicable to the development of other components of real-time technological systems.

Author Contributions: Funding acquisition, V.K.; conceptualization, N.S.; writing—review & editing, E.Z.; writing—original draft preparation, V.N.; software, V.O.; methodology, A.S.; validation, M.S. All authors have read and agreed to the published version of the manuscript.

Funding: The study was financially supported by the Ministry of Science and Higher Education of the Russian Federation under agreement No. 075-15-2022-1024.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Informed consent was obtained from all subjects involved in the study. Written informed consent has been obtained from the patient(s) to publish this paper.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Novikov, V. Solving the problems of intelligent crew support in terms of reconfiguration during OBE in case of failures. In Proceedings of the Collection of Abstracts of the VI International Scientific and Practical Conference "AVIATOR", Voronezh, Russia, 14–15 February 2019; pp. 189–192.
2. Kopetz, H. *Real-Time Systems. Design Principles for Distributed Embedded Applications*, 2nd ed.; Springer: Berlin/Heidelberg, Germany, 2011.

3. Gorbachev, S.; Koblyakova, L.; Sheynin, Y.; Stepanov, A.; Suvorova, E.; Suess, M. Distributed Interrupt Signalling for SpaceWire Networks. In Proceedings of the 5th International SpaceWire Conference, Gothnburg, Sweden, 10–14 June 2013.
4. Koblyakova, L.; Sheynin, Y.; Suvorova, E. Asynchronous hard real time signals transmission in embedded networks. *Int. J. Embed. Real-Time Commun. Syst.* **2014**, *5*, 24–44. [CrossRef]
5. Koblyakova, L.; Gorbachev, S. *Tasks of Hard Real Time Signals in Built-in Systems and Onboard Networks*; SUAI: Saint Petersburg, Russia, 2016; pp. 79–87.
6. Sheinin, Y.; Solokhina, T.; Petrichkovich, Y. SpaceWire technology for parallel systems and onboard distributed systems. *Electron. Sci. Technol. Bus.* **2006**, *5*, 64–75.
7. Zhu, Q.; Yang, Y.; Scholte, E.; Di Natale, M.; Sangiovanni-Vincentelli, A. Optimizing Extensibility in Hard Real-Time Distributed Systems. In Proceedings of the Real-Time and Embedded Technology and Applications Symposium, San Francisco, CA, USA, 13–16 April 2009.
8. Zhu, X.M.; Lu, P.Z. Multi-Dimensional Scheduling for Real-Time Tasks on Heterogeneous Clusters. *J. Comput. Sci. Technol.* **2009**, *24*, 434–446. [CrossRef]
9. Van Steen, M.; Tanenbaum, A. A brief introduction to distributed systems. *Computing* **2016**, *98*, 967–1009. [CrossRef]
10. Aydin, H.; Melhem, R.; Mossé, D.; Mejía-Alvarez, P. Power-aware scheduling for periodic real-time tasks. *IEEE Trans. Comput.* **2004**, *53*, 584–600. [CrossRef]
11. Novikov, V.; Kosyanchuk, V.; Chuyanov, G.; Platoshin, G.; Stetsyuk, S.; Penkin, S. A Real-Time Information Transmission System Based on a Fully Optical Spectrally Multiplexed on-Board Real-Time Network. Patent RU2694137C1, 22 August 2019.
12. Kosyanchuk, V.; Novikov, V.; Kasatikov, M.; Mishchenko, I. Dual-Loop Real-Time Information Transmission System Based on a Fully Optical Wavelength-Multiplexed Onboard Network. Patent RU2744517C1, 11 March 2021.
13. Olenev, V. Methodology for communication protocols formal design based on Petri nets. *Informaciâ i Kosmos* **2022**, *4*, 37–45.
14. Olenev, V.; Suvorova, E.; Korobkov, I.; Rozhdestvenskaya, K.; Chumakova, N.; Sinev, N.; Novikov, V. Software model of the protocol stack functioning as a means of diagnosing the optical onboard network. In Proceedings of the Collection of Reports of the Sixth International Conference "Perspective Directions for the Development of Airborne Equipment of Civil Aircraft", Moscow, Russia, 22 July 2021; pp. 45–56.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.