DOI: https://doi.org/10.15276/eltecs.42.118.2025.6
UDC 004.94:628.4.047

Improvement of automation in managing thermal destruction of municipal solid waste through software recovery from failures

K. Kirkopulo[1], V.Tonkonogyi[1], M. Maksymov[2]
[1]*Odesa Polytechnic National University*
[2]*National University «Odesa Maritime Academy Institute of the Naval Forces»*

Abstract. *The article presents the results of improving automation in managing the thermal destruction of municipal solid waste (MSW) through software recovery after failures. A functional structure and selected technological parameters (as controlled coordinates) for a hierarchical automatic control system (ACS) for MSW thermal destruction were proposed. A homogeneous Markov chain was used to model the failure and fault parameters of cyclic software operation. The application of the ACS with the proposed recovery structure ensures software reliability by adjusting the threshold number of failures. This recovery implementation reduces environmental pollution in stationary modes with known raw material composition, in dynamic modes with material composition changes, and during pyrolysis plant start-up. The proposed approach enhances the accuracy of controlling key technological parameters, operational reliability, and economic performance.*

Keywords*: automation, automatic control system, thermal destruction, municipal solid waste, recovery, software, failure, Markov chain, environment.*

Introduction

Environmental pollution is one of the most pressing modern issues, caused by rapid industrial development and urbanization in many countries. Among the most effective solutions is a fundamentally new technology for municipal solid waste MSW utilization that produces alternative fuel, generally known as thermal destruction. This utilization technology combines circulation pyrolysis and thermally stable gasification into a single process, allowing complete MSW utilization in environmentally safe and energy-efficient modes. This technology also produces gaseous fractions of alternative fuel, which are suitable for use in energy facilities without additional purification [1].

To implement such technology, specialized technological complexes are used, which are complex multicomponent technical objects requiring a specialized hierarchical computerized automatic control system (ACS) [2]. Comprehensive automation of technological complexes for MSW utilization is impossible without highly reliable software, which significantly improves operational efficiency and economic performance.

Addressing the task of improving automation for managing the thermal destruction of MSW should be based on analyzing the main properties and technical characteristics of the technological process with the selection of technological measurement parameters for control adjustments. On the other hand, the complex formalization of chemical-technological process control tasks necessitates increased reliability of ACS software. Despite significant efforts to improve software reliability, such as automated software design systems, libraries of standard procedures, and improved diagnostic tools, errors in software persist. It is believed that completely eliminating software errors is impossible due to the inherent complexity of software systems. The main groups and sources of errors are analyzed in [5], while the consequences of errors are described in [5,6].

An alternative approach to increasing software reliability involves recovering software after failures by utilizing reserves in processing time and computational resources [6].

The goal of this article is to enhance the reliability of ACS software for managing MSW thermal destruction using recovery during the processing of selective measured parameters when adjusting control actions for the input material composition and evaluating the impact of recovery on process efficiency.

1. Fundamental Diagram of a Thermal Destruction Plant for MSW [7].

A typical plant (Figure 1) allows:

In stationary operational modes, where the

composition of the processed raw material is known, to completely gasify its organic components by calculating the minimum required amount of air and adjusting its supply;

– During operation, to monitor the composition of the input material and adapt to changes in control actions by switching to a mode of identifying the composition for restoring management stability;

– In dynamic modes, when the composition of the input material changes, and during the plant's startup, to determine the current composition and enthalpy of the input material.

The schematic of the plant's main part – the Reactor III – is shown in Figure 1. Its operation principles and the values of certain parameters for stationary operation modes correspond to the descriptions provided above. Through Collector I, the input material is supplied to the reactor at a flow rate of G1.

Through Collector II, the hot product gas exits the reactor at a flow rate of Q2, entering the main system and splitting into two streams: Q4 – to the consumer; Q3 – via the recirculation line back into the reactor at point V for drying and heating the input material.

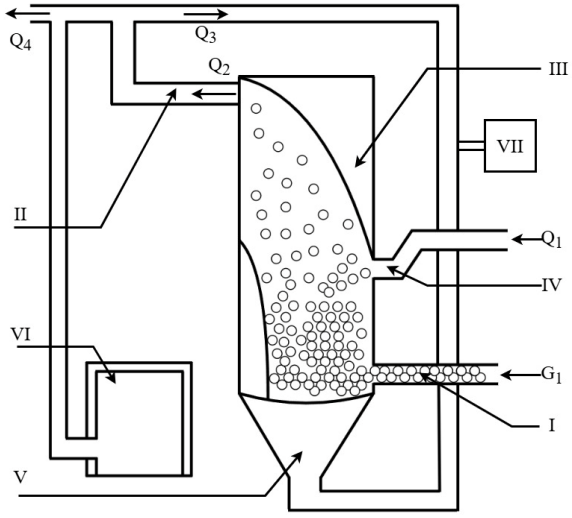

Fig. 1. Schematic diagram of a pyrolysis plant

Through Collector IV, the necessary amount of air is supplied at a flow rate of Q1.

In stationary operational mode and complete gasification of the organic component of the input material, the gross formula of the gas matches that of the input material. The gas composition and its stability are controlled using Device VII, based on the method [8] of determining the gross formula of combustible gas mixtures during their combustion in a specialized device [9]. This process involves measuring the flow rates of the gas mixture, air, and exhaust gases under varying flow rates of the mixture. The gas mixture may include non-combustible gases as well.

If a change in the product gas composition is detected, the plant shifts from the gasification mode to combustion mode. This is achieved by increasing air supply through Collector IV to ensure the input material with an unknown altered composition is completely converted into gaseous products. Subsequently, Device VII determines their gross formula and the new composition of the input material. Based on this data, the plant transitions back to pyrolysis (gasification) mode by adjusting air supply to match the new batch composition.

For the technological implementation of this process, the plant is equipped with approximately 1500 information-measurement channels transmitting no fewer than 20000 individual signals, each carrying parameter values.

2. Development of data collection algorithm

The hardware implementation of the control system for the thermal destruction chemical-technological complex of MSW includes data collection modules and data output interfaces, as well as programmable logic controllers for control. The data collection modules receive analog signals from various sensors and convert them to digital data, which is transmitted to the ACS via RS485 network using the DCON protocol at 115200 baud. Programmable logic controllers perform the functions of automatic regulation of specified values of controlled variables in the chemical-technological process of MSW thermal destruction, as well as transmission to the operator interface implemented on a personal computer.

The output modules, in turn, implement the conversion of digital control signals from programmable logic controllers into analog signals, which directly go to the control elements of the installation's technological parameters.

To receive signals from different types of sensors in this control and monitoring system, the following types of data collection modules are used: thermocouple data collection modules, current output sensor data collection modules (4...20 mA), and discrete input modules. To implement the conversion of digital control signals from PLC to analog signals that directly go to the control elements of the installation's technological parameters, analog output modules are used.

For example, ICP DAS data collection modules have the following operating algorithm (Fig. 2) [10].

The system operation begins with a configuration check procedure, which determines the composition of modules and their addresses. Based on the verification results, appropriate changes can be made. According to the specifics of the tasks being solved, the configuration of each module is deter

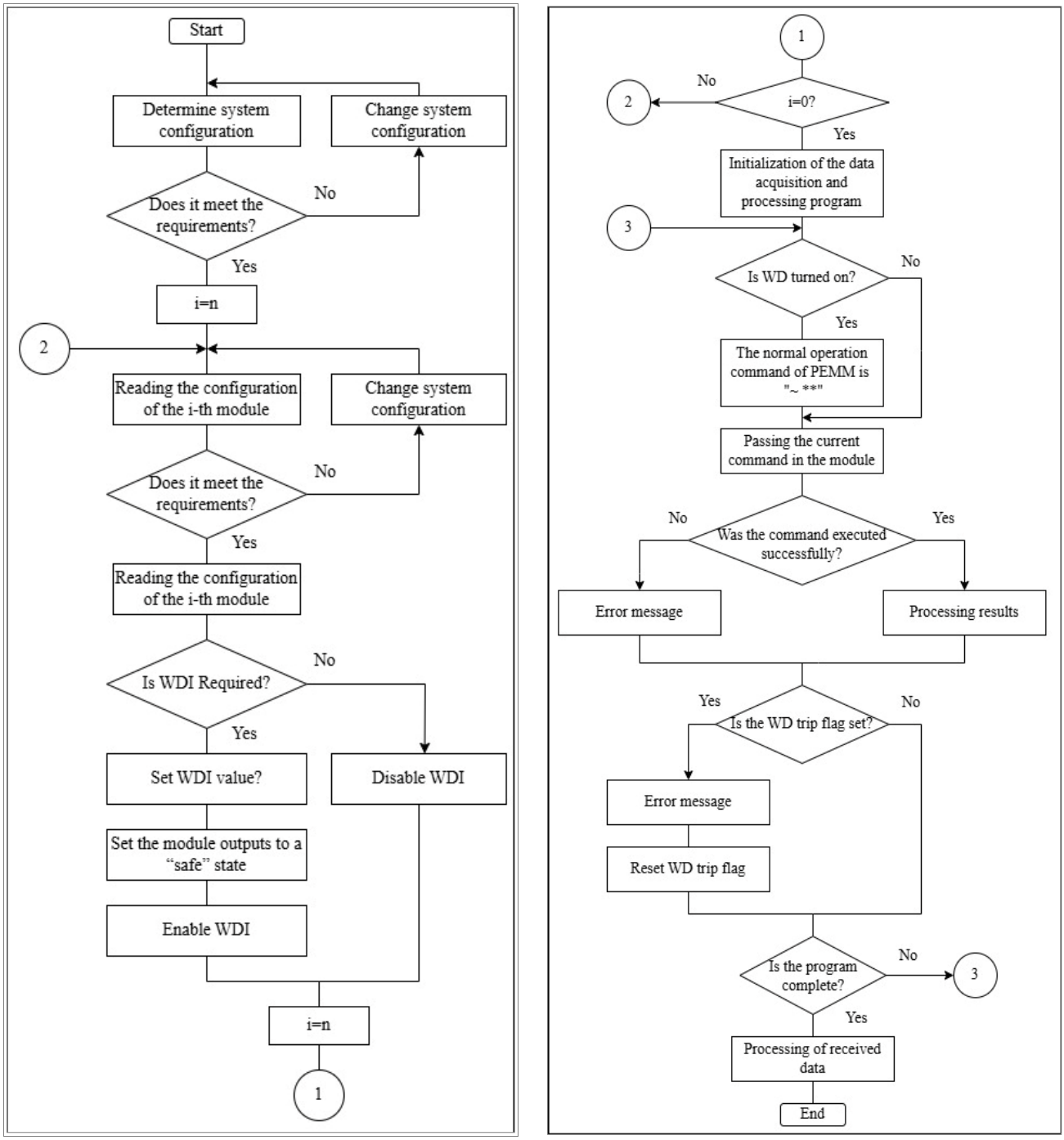

Fig. 2. Block diagram of ICP DAS data acquisition modules

mined, which is controlled through the user interface.

Depending on the system operating conditions, a decision is made about using the watchdog timer (WD) and setting its parameters. After configuring the modules, a data collection and processing program is developed. When using WD, the watchdog timer is reset in each cycle of current command execution.

Command processing by the module includes sequential verification of the following parameters: RS-485 interface exchange rate; operation code; module address; checksum analysis, if provided; operation execution correctness; frame completion correctness.

To increase information reliability, «Set Value» and «Check Value» commands are used. However, with increased speed requirements, such checks can be minimized.

Monitoring the WD trigger flag helps prevent system «hanging». Depending on the algorithm's specifics, data processing can be performed after executing each command and/or at the end of the information collection procedure (for example, when building graphs).

Input information reading from modules can be performed in asynchronous or synchronous modes. Synchronous mode is used when it is necessary to temporarily fix input parameter values in several modules.

To implement synchronous mode, a synchronized sampling command should be used, by which modules supporting this mode store input information in internal registers. Then, information from these registers is read asynchronously.

The hardware for measuring basic technical parameters of the ash utilization technological complex includes various sensors of the complex's operating parameters, including: temperature sensors - thermocouples of two types: TXK - for measuring temperatures in the range of -40...600°C, and TXA - for measuring temperatures in the range of -40...1050°C; level sensors - float type - for level measurements, and radar type - for measuring raw material level in the pyrolysis reactor; pressure sensors; gas and liquid flow sensors and others. Analysis of the implemented software architecture (TRACE MODE 6) showed satisfactory values of 3 to 5 failures per eight-hour operator shift. Subsequently, the task arises to develop a methodology for calculating such a value.

3. Method for Improving the Reliability of ACS Software for the Thermal Destruction of Solid Waste

A failure is understood as a disruption in the functionality of the software for a duration shorter than a threshold value. Conversely, a fault refers to a disruption in software functionality that exceeds this threshold. If no recovery actions are taken, all failures effectively become faults. The threshold recovery time is determined by the nature of the Automatic Control System (ACS).

Let us consider the organization of the software structure of the ACS for the thermal destruction of solid waste. This software operates in a cyclic mode, meaning that the following sequence of actions is repeatedly executed (see Figure 2): polling sensors and receiving commands from the operator or other systems; analyzing and processing sensor readings and generating control actions; and finally, executing the control actions.

Sensor readings and operator commands serve as input data. Operator commands and signals from other systems are usually strictly formalized and undergo verification before execution, allowing invalid commands that could disrupt the software's functionality to be discarded. However, sensor readings are subject to inaccuracies from the sensors themselves and their converters, as well as random noise in communication channels, which can lead to significant distortion of readings.

During the development of ACS software, debugging and testing are performed. The most critical software functions undergo a verification procedure that defines the process for conducting checks and documenting identified deviations. However, verification is carried out on a limited set of initial data selected during software design and cannot guarantee absolute fault-free operation. Therefore, debugging and testing are conducted on a finite set of initial data—control examples. Based on the results of debugging and testing, one can confidently assert the software's fault-free operation for these control examples, but its reliability for other initial data remains uncertain.

Meanwhile, during software operation, input data combinations that were not included in the control examples may occur, leading to failures. In most cases, software functionality can be restored by simply skipping the cycle in which the failure occurs and processing new data in the next cycle. However, if failures occur in several consecutive cycles, this is classified as a fault. The maximum allowable number of failures before a fault occurs, N, is determined by the nature of the ACS and the available time and processing speed reserves within a single software cycle and can be derived from the threshold recovery time.

A fault is defined as the consecutive occurrence of failures over n execution cycles of the program. The threshold time that distinguishes a fault from a failure is equal to the duration of n processing cycles. For software without recovery, which will be used for comparison, $n = 1$.

Experience shows that the operation of the software in each cycle is independent of the results of previous executions, and the input data are updated in each cycle. Let us assume that the probability of failure in a single execution cycle of the program is constant and equal p. We will calculate the probability of software failure over a sufficiently large but finite number of cycles.

For this calculation, we use a simple homogeneous Markov chain, as shown in Fig. 3.

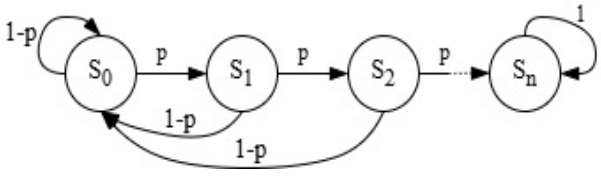

Fig. 3. Homogeneous Markov Chain, $n > 1$

The state S_0 corresponds to the normal execution of the current software operation cycle. If the first failure occurs at N_f, the system transitions to

state S_1. The probability of this transition is p. In the next cycle, a second failure may occur with probability p (transitioning to state S_2), or with probability $1 - p$, no failure will occur, and the system will return to state S_0. A fault corresponds to state S_n, which the software reaches after n consecutive failures. Transitions from state S_n to other states are not possible. We assume that once the software reaches state S_n, it remains in that state with probability 1. For the case where $n = 1$ (software without recovery), the Markov chain has the structure shown in Fig. 4.

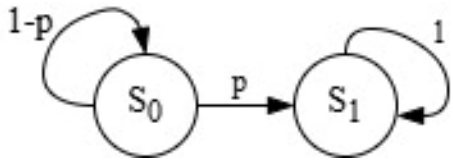

Fig. 4. Homogeneous Markov Chain, $n = 1$

The transition probabilities between states in these Markov chains are described using transition probability matrices. The matrices for the cases $n = 1$, $n = 2$, $n = 3$, and $n = 4$ are provided below.

$$R_1 = \begin{vmatrix} 1-p & p \\ 0 & 1 \end{vmatrix};$$

$$R_2 = \begin{vmatrix} 1-p & p & 0 \\ 1-p & 0 & p \\ 0 & 0 & 1 \end{vmatrix};$$

$$R_3 = \begin{vmatrix} 1-p & p & 0 & 0 \\ 1-p & 0 & p & 0 \\ 1-p & 0 & 0 & p \\ 0 & 0 & 0 & 1 \end{vmatrix};$$

$$R_4 = \begin{vmatrix} 1-p & p & 0 & 0 & 0 \\ 1-p & 0 & p & 0 & 0 \\ 1-p & 0 & 0 & p & 0 \\ 1-p & 0 & 0 & 0 & p \\ 0 & 0 & 0 & 0 & 1 \end{vmatrix}.$$

The initial probability is $P_{S_0}^0 = 1$. For all other states, the initial probabilities are zero.

The software failure probability P represents the probability of transitioning from the initial state S_0 to state S_1. This corresponds to the conditional probability P, which is found in the last column of the first row (the upper-right corner of the transition probability matrix). At the beginning of the software operation (in the «zero» cycle), this probability is zero. To determine the probability of failure at the N-th cycle, the transition matrix must be raised to the N-th integer power R_1.

For practical calculations, instead of using 1, a probability value of $P(S_0)^0 = 0.999$ can be used. This probability is reached at a certain critical cycle number N_{crit}, meaning that a software failure will occur at a cycle number less than or equal to N_{crit}. The calculations were performed for $N = 10^{10}$ independent cycles.

The calculation of Markov matrices allows for the construction of dependencies between the software failure probability, the cycle number, and the failure probability per cycle. For practical applications, it is more convenient to use inverse dependencies of failure probabilities leading to a fault as a function of the cycle number.

For the considered cases, Table 1 provides the values of the coefficients (a and b) for $n = 1$, $n = 2$, $n = 3$, and $n = 4$, using the inverse dependency equation of the form: $p(n) = a(n)N + b(n)$

Calculations using this equation allow for determining the probability of failure in a single cycle, which is required to achieve the desired maximum number of execution cycles N before failure.

Table 1 shows the values of the coefficients (a and b) for predicting the maximum number of software execution cycles N before failure depending on the probability of failure per cycle p and the maximum permissible number of failures n before failure.

Table 1

Values of the coefficients for predicting the number of software execution cycles

n	a	b
1	-7,69E-03	1,50E-01
2	-7,76E-04	1,15E-01
3	-9,54E-05	1,05E-01
4	-1,25E-05	1,02E-01
5	-1,67E-06	1,01E-01

To increase the number of failure-free software execution cycles, it is necessary to either reduce the probability of failure per cycle or increase the maximum allowable number of failures before a fault. Table 1 presents the case for $n = 5$, obtained through an approximation of the calculated results.

To convert the number of cycles into program execution time and mean time to failure (MTTF), it is sufficient to multiply the cycle number by the cycle duration. The ACS software for thermal destruction of solid waste employs a recovery mechanism with a maximum allowable number of failures $n = 3$. The duration of a single cycle is 0.1 seconds.

During software operation, individual failures occurred at the following cycle numbers: 54, 106, 156, 208. The average interval between failures is N_f = 52. The average probability of failure per cycle is p = 0.192. Using the given formula and values from Table 1, we find that $N_{crit} = 9 \times 10^5$, which corresponds to 25 hours of software operation.

This significantly increases the maximum failure-free operation time of the software with recovery, compared to software without recovery.

Conclusions

The application of a recovery mechanism makes it possible to develop software with a specified level of reliability by adjusting the maximum allowable number of failures before a fault occurs.

The implementation of software recovery support is most conveniently achieved through the use of a dedicated library. Such a library serves as an intermediary between application software, the operating system, and automation hardware. At the library level, the processing cycle is organized, including the reception of input data and the issuance of control actions.

In addition, the library may contain auxiliary functions, such as archiving and logging, which provide solutions to two important tasks.

First, it enables the identification of specific input data combinations that lead to failures, which in turn facilitates the correction of errors in the software.

Second, it allows for the determination of the failure-free operating time of the software, making it possible to take preventive measures to eliminate detected errors before a fault occurs.

References

1. Ryzhkov, S.S. and Markina, L.M., 2007. Experimental studies on the utilization of organic waste by multiloop circulation pyrolysis. *Scientific Journal of the National University of Shipbuilding*, (5), pp.100–106.

2. Kondratenko, Y.P. and Kozlov, O.V., 2011. Analysis of task complexes and control coordinates of the ecopyrogenesis technological process. *Technical News*, 1(33), 2(34), pp.13–16. DOI https://doi.org/10.1007/978-3-642-30433-0_18

3. Kondratenko, Y.P. and Kozlov, O.V., 2012. Functional structure and computer components of a control system for a multiloop pyrolysis plant. *Bulletin of the National University of Shipbuilding*, 2011 edition. Mykolaiv: NUS, pp.413–423.

4. Brooks, F.P., 1995. *The Mythical Man-Month: Essays on Software Engineering*. Anniversary Edition. Chapel Hill: University of North Carolina. Addison-Wesley, 304p.

5. Maysyan, I.G., 2003. Methods for improving the reliability of software in automated control systems. *Proceedings of the Odessa Polytechnic University*, 2(20), pp.117–120.

6. Maksymov, M.V., Maysyan, I.G. and Kalnev, L.L., 2006. Determination of the probability distribution of software failures in ACS TP. *Proceedings of the Odessa Polytechnic University*, 2(26), pp.101–104.

7. Butenko, O.V., Maksymov, M.V., Demydenko, V.E. and Brunetkin, O.I., 2019. Method for solving complex interpretation problems to address the inverse task of determining fuel composition. *Ship Power Plants: Scientific-Technical Collection*, (39), pp.30–42. Available at: http://seu.onma.edu.ua/wp-content/uploads/2020/09/2019_39.pdf [Accessed 9 Jan. 2025].

8. Maksymov, M.V., Brunetkin, A.I. and Bondarenko, A.V., 2013. Model and method for determining the conditional formula of hydrocarbon fuel during combustion. *Eastern-European Journal of Advanced Technologies*, 6(8)(66), pp.20–27.

9. Patent of Ukraine No. 120216, 2019. Device for determining the composition of combustible gas during its combustion. Filed 22 Dec. 2017, Published 25 Oct. 2019. *Bulletin*, (20).

10. Jang, J.S.R., Sun, C.T. and Mizutani, E., 1996. *Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence*. Prentice Hal

Вдосконалення автоматизації управління термічним знищенням твердих побутових відходів через відновлення програмного забезпечення після збоїв

К. Г. Кіркопуло[1], В. М. Тонконогий[1], М. М. Максимов[2]

[1]Національний університет «Одеська політехніка»

[2]Національний університет «Одеська Морська Академія Інститут Військово-Морських Сил»

Анотація. *У статті представлені результати вдосконалення автоматизації управління процесом термічного знищення твердих побутових відходів (ТПВ) шляхом відновлення програмного забезпечення після збоїв. Запропоновано функціональну структуру та обрано технологічні параметри (як керовані координати) для ієрархічної автоматичної системи управління (АСУ) процесом термічного знищення ТПВ. Для моделювання параметрів збоїв та відмов у циклічному функціонуванні програмного забезпечення використано однорідний Марковський ланцюг. Застосування АСУ із запропонованою структурою відновлення забезпечує надійність програмного забезпечення шляхом регулювання порогового значення кількості збоїв. Впровадження цього підходу зменшує забруднення довкілля у стаціонарних режимах із відомим складом сировини, у динамічних режимах із зміною складу матеріалів, а також під час запуску піролізних установок. Запропонований підхід підвищує точність контролю ключових технологічних параметрів, надійність роботи та економічну ефективність.*

Ключові слова: *автоматизація, автоматична система управління, термічне знищення, тверді побутові відходи, відновлення, програмне забезпечення, збій, Марковський ланцюг, довкілля.*

Отримано 20.01.2025

About the authors

Katerina G. Kirkopulo, Doctor of Philosophy (Ph.D.), Associate Professor of the Department of Information Technology, Engineering and Design, Odesa Polytechnic National University, 1, Shevchenko ave., Odesa, 65044, Ukraine.
E-mail: kirkopulo.k.g@op.edu.ua; ph.: +38 093 411 1013

Кіркопуло Катерина Григорівна, доктор філософії (Ph.D.), доцент кафедри інформаційних технологій проектування та дизайну, Національний університет «Одеська політехніка»; проспект Шевченка, 1, Одеса, 65044, Україна.
E-mail: kirkopulo.k.g@op.edu.ua; тел. +38 093 411 1013

ORCID: 0000-0001-5570-5989

Vladimir M. Tonkonogyi, Dr. of Technical Sciences, Professor, Director of the Institute of Digital Technologies, Design and Transport, Odesa Polytechnic National University;
1, Shev-chenko Ave., 1, Odesa, 65044, Ukraine.
E-mail: vmt47@ukr.net; ph.: +38 067 484 1287

Тонконогий Володимир Михайлович, д. т. н., професор, директор інституту цифрових технологій, дизайну та транспорту, Національний університет «Одеська політехніка»; проспект Шевченка, 1, Одеса, 65044, Україна.
E-mail: vmt47@ukr.net; тел. +38 067 484 1287

ORCID: 0000-0003-1459-9870

Maksym M. Maksymov, senior researcher, NDC of the Armed Forces of Ukraine "State Oceanarium" of the Institute of the Naval Forces, National University "Odesa Maritime Academy"; 8, St. Didrichson, Odesa, Ukraine.
E-mail: maximov.agro@gmail.com; ph.: +38 093 970 2121

Максимов Максим Максимович, старший науковий співробітник, НДЦ ЗСУ "Державний океанаріум" Інституту Військово-Морських Сил, Національний університет «Одеська морська академія»; вул. Дідріхсона, 8. Одеса, Україна.
E-mail: maximov.agro@gmail.com; тел. +38 093 970 2121

ORCID: 0000-0002-5626-5265