Международный журнал информационных технологий и энергоэффективности

Том 8 Номер 8 (34)

2023

Международный журнал информационных технологий и энергоэффективности
Т.8 №8(34) 2023 г.

Сайт журнала:
http://www.openaccessscience.ru/index.php/ijcse/

СОДЕРЖАНИЕ / CONTENT

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

Коннов Д.В. Реализация арифметического калькулятора на языке OBJECT PASCAL // Международный журнал информационных технологий и энергоэффективности. – 2023. – Т. 8 № 8(34) с. 4–16

Международный журнал информационных технологий и энергоэффективности

Сайт журнала:
http://www.openaccessscience.ru/index.php/ijcse/

УДК 004.9

РЕАЛИЗАЦИЯ АРИФМЕТИЧЕСКОГО КАЛЬКУЛЯТОРА НА ЯЗЫКЕ OBJECT PASCAL

Коннов Д.В.
Университет Центральной Флориды, Орландо, США (32816, Орландо, Бульвар Центральная Флорида, 4000), e-mail: konnov72@knights.ucf.edu

В статье представлен обзор основных принципов построения арифметического калькулятора, включающего компьютеризированные расчеты по алгоритму маневровой станции и польской нотации PRN. Предложена программная реализация будущего облачного инженерного калькулятора на примере JclExprEval.pas из библиотеки кода JEDI (JCL), одной из наиболее поддерживаемых и актуальных библиотек языка Object Pascal на сегодняшний день. Рассмотрены вопросы токенизации входного выражения, построения дерева разбора, рекурсивного спуска и вычисления дерева. Отмечены требования, необходимые для реализации качественного продукта и приведены примеры реализации кода. Дано обоснование преимуществ использования калькулятора в инженерных расчетах в составе ядра технической системы. На основе этого исследования предложены пути и подходы к разработке будущего многопользовательского системного калькулятора. Обоснована новизна исследования как направление интеграции калькулятора с облачными системами и отделение скриптовых формул приложения от компилируемого кода.

Ключевые слова: Математический эвалюатор, JEDI (JCL), Delphi, токены, дерево синтаксического анализа, дерево синтаксического анализа, синтаксический анализ выражений, рекурсивный спуск, алгоритм сортировочной станции, польская нотация PRN, стек.

IMPLEMENTATION OF ARITHMETIC CALCULATOR IN OBJECT PASCAL

Konnov D.V.
University of Central Florida, Orlando, USA (4000 Central Florida Blvd., Orlando, 32816), e-mail: konnov72@knights.ucf.edu

The article provides an overview of the fundamental principles of constructing an arithmetic calculator including computerized calculations Shunting Yard algorithm and Polish notation PRN. Proposed a software implementation of a future Cloud-based engineering calculator using the JclExprEval.pas as an example from the JEDI code library (JCL), one of the most supported and relevant libraries of the Object Pascal language nowadays. The issues of tokenization of the input expression, building a parse tree, and recursive descent and evaluation of the tree are considered. The requirements necessary for the implementation of a quality product are noted and examples of code implementation are given. The rationale for the advantages of using a calculator in engineering calculations as part of the core of an engineering system is given. Proposed ways and approaches for the development of the future multiuser system calculator based on this research. The novelty of the research is substantiated as a direction of integration of the calculator with Cloud systems and separation of application scripted formulas and compile code.

Keywords: Evaluator, JEDI (JCL), Delphi, tokens, parse tree, Parsing Tree, expression parsing, recursive descent, Shunting Yard Algorithm, Polish notation PRN, stack.

The Problem.

Building an arithmetic calculator involves implementing an algorithm that can calculate mathematical expressions using operations such as addition, subtraction, multiplication, division, and parentheses. Let us immediately define this concept. A calculator is a software device that receives an arithmetic or algebraic expression as a string as input. This apparatus has an interface for adding or determining the values of variables used in the evaluated expression, as well as events that are called during the parsing of the expression, where the values of variables can be substituted dynamically. The result of the Calculator is a floating-point value, i.e., the result of calculating the expression given as input. Let's consider the main stages of developing an arithmetic expression calculator in the Delphi environment using the example of the Evaluator from the JEDI library (JCL) [1], and then we will analyze the detailed aspects of the internal implementation of the parser.

To start, a little history:

The Shunting Yard algorithm [2] is a classic method for parsing infix mathematical expressions and converting them to reverse Polish notation (RPN) [3], also known as postfix notation. RPN is a format in which operators are placed after their operands, making it easier to evaluate expressions using the stack approach.

The algorithm was introduced by Edsger W. Dijkstra [4] in 1961 and is commonly used in compilers, interpreters, and calculator programs. The key idea of the Shunting Yard algorithm is to use two stacks [5], one for operators and one for output (RPN) values, to process an expression respecting operator precedence and associativity.

Here is a step-by-step explanation of how the shunting station algorithm works:

1. While there are tokens to be read:
2. Read a token
3. If it's a number add it to the queue
4. If it's an operator
5. While there's an operator on the top of the stack with greater precedence:
6. Pop operators from the stack onto the output queue
7. Push the current operator onto the stack
8. If it's a left bracket push it onto the stack
9. If it's a right bracket
10. While there's not a left bracket at the top of the stack:
11. Pop operators from the stack onto the output queue.
12. Pop the left bracket from the stack and discard it
13. While there are operators on the stack, pop them into the queue

The resulting RPN expression can be evaluated using the stack approach, in which the operands are placed on the stack, and when an operator occurs, the required number of operands are retrieved from the stack and the result of the operation is placed back on the stack.

The Shunting Yard algorithm ensures that the RPN expression maintains the correct order of operations based on operator precedence and associativity and allows the expression to be evaluated efficiently without using parentheses for grouping.

Let's convert the expression: "(2 + 3) * 4" to RPN

Step 1: Initialize an empty stack for statements and an empty queue (or list) for output (RPN) values (operations are presented in Table 1).

Step 2: Iterate over the tokens of the input infix expression "(2 + 3) * 4":

Table 1 – Step 1 operations

Input Token	Stack (operators)	Output Queue (RPN)
(	(	
2	(	2
+	+ (	2
3	+ (	2 3
)		2 3 +
*	*	2 3 +
4	*	2 3 + 4

Step 3: Extract all remaining statements from the statement stack and add them to the output queue, the operator (*) remains the last on the stack:

Table 2 – Step 3 operations

Input Token	Stack (operators)	Output Queue (RPN)
		2 3 + 4 *

The RPN expression – "2 3 + 4 *" is the equivalent of the original infix expression "(2 + 3) * 4" in reverse Polish notation (RPN). In RPN, operators come after their operands, and parentheses are not needed for grouping, because the order of operations is implicitly determined by the position of the operators in the expression.

To evaluate the RPN expression "2 3 + 4 *", a stack-based approach can be used. As follows:

1. Initialize the empty stack.
2. Iterate over an RPN expression from left to right.
3. If the token is a number (operand), then it is placed on the stack.
4. If the token is an operator, then the required number of operands is extracted from the stack, an arithmetic operation is performed, and the result is placed back on the stack.
5. Steps 3 and 4 continue until all tokens have been processed.
6. After processing all the tokens, the result will be at the top of the stack.

Evaluate the RPN expression "2 3 + 4 *" (operations are presented in Table 3):

Table 3 – Operations

Token	Stack
2	2
3	2 3
+	5
4	5 4
*	20

After processing all the tokens, the result in the stack is 20. Therefore, the actual result of the expression "2 3 + 4 *" is 20.

Materials and research methods

The Object Pascal language in modern days has proven itself as a reliable, high-level programming language that allows you to build systems on the scale of large corporations and enterprises. The development of an arithmetic engineering calculator [6] in this language is a completely natural and essential task of any corporate software product in the Delphi environment [7].

The JclExprEval.pas module in the JEDI Code Library (JCL) provides a flexible *expression evaluator* capable of analyzing and evaluating mathematical expressions. It allows you to dynamically evaluate arithmetic, Boolean, and comparison expressions at runtime. Let's take a closer look at how the JclExprEval.pas parser works. The main functionality of JclExprEval is to parse the input expression, building a *parsing tree* (Parse Tree) [9] and its evaluation. Below are abstracted implementation details that may not be properly aligned with the latest library code but with the purpose of explaining how it all works.

First, the input expression must be tokenized.

Tokenization: This is the division of an input mathematical expression into individual tokens, including numbers, operators, and parentheses [10]. The expression is first tokenized by breaking it down into individual components, such as numbers, operators, functions, and variables. Tokens are the building blocks that the analyzer uses to understand the expression when building a *parse tree*. Each token is represented by a **TExprToken** record that contains information about its type, value, and position in the expression.

The TExprToken record is a fundamental data structure used in the JclExprEval.pas module to represent individual tokens in the parsed expression. It contains information about the type, value, and position of the token in the original expression. The definition of the TExprToken record is shown in Figure 1.

```
type
  TExprToken = record
    TokenType: TExprTokenType;
    TokenValue: Variant;
    Position: Integer;
  end;
```

Figure 1 – Structure of the expression token.

1. TokenType: The TokenType field specifies the type of token, which can be one of the following values (defined in the JclExprEval.pas module):

- etUnknown: Unknown type.
- etNumber: Numeric value.
- etVariable: Variable name.
- etOperator: operator.
- etLeftParenthesis: left parenthesis "(".
- etRightParenthesis: right parenthesis ")".
- etFunction: A token that represents the name of the function.

2. TokenValue: The TokenValue field contains the actual value of the token. Its data type is Variant, which means that it can store different types of values depending on the type of token:

- etNumber: numeric value in the form of Double.
- etVariable: variable name as a string.
- etOperator: operator as a string.
- etFunction: function name as a string.

3. Position: The Position field specifies the position of the token in the original expression as an index. The index is zero-based, which means that the first character in the expression has position 0.

Each token is part of an expression, including numbers, operators, and parentheses, and stores the corresponding properties in a TExprToken record. TExprToken entries are temporarily stored in an internal array in the order in which they appear in the expression. This array of TExprToken entries is used to represent the tokenized expression during the parsing step.

For example, the input expression "2 + 3 * 4" is tokenized, and TExprToken entries are created for each token. The resulting array of TExprToken records might look like this:

[Token(2), Token(+), Token(3), Token(*), Token(4)]

Each element of this array is a TExprToken entry. This is important for the parsing and evaluation process because the Token helps the JclExprEval module understand the structure of the expression and perform the necessary operations to evaluate it. Next, the analyzer starts parsing the token array. When the algorithm encounters each token, it uses the information stored in the token to build the structure of the parse tree and make decisions about how to handle subsequent tokens. Depending on the type of token, the algorithm creates operator nodes, operand nodes, or functional nodes. In the process of processing tokens, the algorithm builds **TJclExprNode** objects and collects these nodes (nodes) into a *parsing tree* (Figure 4).

The structure of TJclExprNode is shown in Figure 2. In JclExprEval.pas, each node *in the parse* tree is represented by a specific class named TJclExprNode. The TJclExprNode class is the base class for various types of nodes in the parser tree, including operator nodes, operand nodes, and functional nodes. TJclExprNode (the base class for all nodes) represents a generic node in the parse tree. Contains properties and methods that are common to all types of nodes. The TJclExprNode class and its subclasses in the parse tree support referencing child nodes through their FLeft and FRight properties. The parse tree is built using these references, which allows nodes to be linked to each other hierarchically. The TJclExprNode class has properties that define the structure of the parse tree and how the nodes are connected:

```
type
  TJclExprNode = class(TObject)
  private
    FLeft: TJclExprNode;   // Reference to the left child node (if any).
    FRight: TJclExprNode;  // Reference to the right child node (if any).
    // Other fields and methods specific to the base class
  public
    // Public methods and properties, common to all node types.
  end;
```

Figure 2 – The structure of the node of the parsing tree.

FLeft: This property contains a reference to the left child node of the current node. It allows for a hierarchical organization of nodes, where the left child node is usually the operand, argument, or first operand for the binary operator.

FRight: This property contains a reference to the right child node of the current node. It is used for binary operators, where the right child represents the second operand.

In subclasses of TJclExprNode (Figure 3), Certain types of nodes can extend these properties as needed to meet their specific requirements.

```
type
  TJclExprOpNode = class(TJclExprNode)
  private
    FOperator: string;  // The operator symbol (e.g., '+', '-', '*', '/')
    // Other fields and methods specific to operator nodes.
  public
    // Additional properties and methods specific to operator nodes.
  end;

type
  TJclExprConstNode = class(TJclExprNode)
  private
    FValue: Double;     // The numeric value of the constant.
    // Other fields and methods specific to constant nodes.
  public
    // Additional properties and methods specific to constant nodes.
  end;
```

Figure 3 – An example of declaring classes derived from TJclExprNode nodes.

JclExprEval.pas implements the TJclExprOpNode node class and its descendants, each of which performs a specific arithmetic function, for example:

- assignment statement "="
- Boolean operators (e.g., AND, OR)
- relational operators (e.g., >, <, =)
- побитовые операторы (например, AND, OR, XOR)
- shift operators (e.g., SHL, SHR)
- addition operator "+"
- subtraction operator "-"
- Multiplication operator "*"
- division operator "/"
- operator obtaining the remainder by division "%" (mod)
- unary operators (e.g., unary minus)
- Degree Operator "^"
- factorial operator "!"

TJclExprNode also has the following subclasses:

- TJclExprVarNode: variable.
- TJclExprConstNode: constant (numeric value).
- TJclExprFuncNode: represents a function call. Contains properties for storing the function name and argument node references.

Parsing Tree

The next step in processing a tokenized expression is to build a parsing tree (Figure 4). This is the process of constructing a hierarchical representation of the structure of an expression. Expression nodes, put together using **FLeft, Fright** pointers, form a *parsing tree*. A tree is a tree-like data structure in which each node represents an operator or function, and its child elements represent operands or arguments. In the JclExprEval.pas module of the JEDI Code Library (JCL), the recursive descent algorithm **does not use the stack to build the parse tree.** The algorithm directly builds the parse tree, recursively [11] evaluating the expression based on the rules of grammar and the precedence of operators. This is implemented using a recursive function called ParseExpression. This function is responsible for the recursive evaluation of the input expression and the construction of the parse tree. The following is the declaration of the ParseExpression function.

```
function ParseExpression(const Expression: string; Variables: TJclExprVariables = nil;
  Functions: TJclExprFunctions = nil; const ParserOptions: TJclExprParserOptions = [];
  const CustomOptions: TJclExprCustomOptions = []): TJclExprNode;
```

Figure 4 – ParseExpression function declaration.

Function parameters:

- Expression: Input expression for analysis and evaluation.
- Variables: A set of variables that can appear in an expression (optional).
- Functions: A set of user-defined functions that can be used in an expression (optional).
- ParserOptions: additional parameters that control parsing behavior.

- CustomOptions: More configurable options to enhance parsing capabilities.

Return value:

- TJclExprNode: The root *of the parse tree* that represents the input expression.

The ParseExpression function is the entry point for starting the recursive descent process for parsing and building the parse tree. It performs recursive parsing and evaluation of the expression, creating the appropriate nodes and linking them together to form a hierarchical parse tree structure. As soon as the ParseExpression function is called with an input expression, it returns the root of the parse tree, which can later be used for evaluation or other operations on the expression.

For example, with the expression "2 + 3 * 4", the parse tree might look like this:

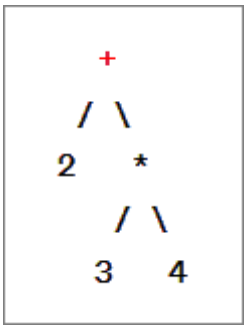


Figure 5 – Parsing tree.

In this example, the + operator node has 2 as the left child node and the * operator node as the right child node. The * operator node, in turn, has 3 as the left child node and 4 as the right child node.

It is easy to see: The nodes of the OPERANDS do not point to any other nodes in the parsing tree, and the nodes of the OPERATORS and FUNCTIONS point to the nodes of the OPERANDS.

Let's summarize the algorithm for constructing a parsing tree:

1. Tokenization: The input expression is tokenized, and each token (e.g., numbers, operators, functions, variables) is represented by a TExprToken record.

2. Building a Parse Tree: Once tokenization is complete, the Recursive Descent algorithm begins to build the parsing tree. It recursively processes tokens, evaluating the expression based on grammar rules and operator precedence.

3. Node creation and linking: When the recursive descent algorithm encounters each TExprToken, it creates the corresponding nodes and links them together as children and parents to form the parse tree structure. The algorithm sets the FLeft and FRight properties of operator nodes based on their operands to properly link them together.

4. Operator precedence: The parsing algorithm considers the priority of operators to ensure that expressions are evaluated according to the correct order of operations (for example, multiplication before addition).

5. Associativity: For operators with the same priority, the parsing algorithm considers their associativity (left-to-right or right-to-left) to maintain the correct order of calculation.

6. Hierarchical relationships: The recursive descent process provides a hierarchical organization of nodes in the parse tree based on the structure of the expression.

7. Root node: After the recursive descent process is complete, the root node of the parse tree represents the entire expression, and the parse tree is ready for evaluation.

Evaluation. Building a parse tree is an intermediate step in the overall process. To get the final result of the expression, the algorithm needs to evaluate the constructed *parsing* tree. The evaluation process involves traversing the tree and applying mathematical operations defined by operators and functions. The algorithm will recursively evaluate subexpressions and combine their results based on the operators and functions encountered during the traversal.

For example, consider the expression "2 + 3 * 4" (Figure 5):

To evaluate the expression, the algorithm will run at the root of the parse tree (the + operator). It will recursively traverse the tree, evaluating the left and right subtrees. The algorithm will first evaluate the operator node * (3 * 4 = 12) and replace the operator node * with its result (12). The updated tree is shown in Figure 6.

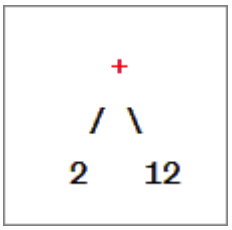


Figure 6 – The Parsing Tree in the process of parsing.

The algorithm will now evaluate the operator node + (2 + 12 = 14) and replace the operator node + with its result (14).

The result of the expression is 14.

It should be noted that to ensure the quality of the calculator, the correct handling of errors of invalid expressions, division by zero, inappropriate parentheses, etc., must be implemented, and support for additional functions such as trigonometric functions, exponentiation, variables, and much more can be added (Figure 7).

```
procedure Init(Evaluator: TEasyEvaluator; FuncList: TStrings);
begin
  with Evaluator do
  begin
    // Constants
    AddConst('Pi', Pi);

    // Functions
    AddFunc('LogBase10', LogBase10);
    AddFunc('LogBase2', LogBase2);
    AddFunc('LogBaseN', LogBaseN);
    AddFunc('ArcCos', ArcCos);
    AddFunc('ArcCot', ArcCot);
    AddFunc('ArcCsc', ArcCsc);
    AddFunc('ArcSec', ArcSec);
    AddFunc('ArcSin', ArcSin);
    AddFunc('ArcTan', ArcTan);
    AddFunc('ArcTan2', ArcTan2);
    AddFunc('Cos', Cos);
```

Figure 7 – Registration of user-defined functions.

Also, the evaluator implements the ability to register variables (Fig. 8) and provides an interface for their initialization.

```
procedure TExprEvalForm.FormCreate(Sender: TObject);
begin
  FEvaluator := TEvaluator.Create;
  FEvaluator.AddVar('x', FX);
  FEvaluator.AddVar('y', FY);
  FEvaluator.AddVar('z', FZ);
  Init(FEvaluator, FuncList.Items);
end;
```

Figure 8 – Registration of Evaluation variables.

Implementing the User Interface

An illustrative example of the user interface (Figure 9.), which allows you to enter arithmetic expressions, and evaluate and display the results can be found in the examples\common\expreval folder of the JEDI library (JCL). An example of using the Calculator is shown in Figure 10.

Figure 9 – Graphical user interface of the Calculator.

```
function ResultAsText(Evaluator: TEvaluator; const Input: string): string;
begin
  try
    Result := FloatToStr(Evaluator.Evaluate(Input));
  except
    on E: Exception do
      Result := E.Message;
  end;
end;
```

Figure 10 – An example of using the Evaluator.

Discussion of the results

One of the main requirements for engineering computing systems these days is, of course - performance. It is worth noting that the performance of this calculator on the local Desktop system is quite satisfactory and does not raise questions from the user. However, the implementation of such a calculator as part of a multi-user system can cause performance problems [12]. Therefore, the developer will have to think about the implementation of the system parallelism [13] and base further development on its principles.

To improve calculation speed, we can consider applying the following approaches:

1. **Reduce Expression Complexity**: Complex expressions with many nested functions and operators can lead to slower evaluation. We want to simplify expressions where possible to reduce the number of operations.
2. **Precompile Expressions**: If we have expressions that are evaluated frequently, consider precompiling them into a reusable format. This can save time on parsing and tokenizing the expression each time it's evaluated.
3. **Caching**: If our calculations involve the same set of variables but different expressions, we may consider caching the values of variables that don't change often. This can reduce the overhead of variable lookup.
4. **Reuse Instances**: If we need to evaluate multiple expressions in the same context, we must reuse the same instance of the expression evaluator rather than creating a new one for each evaluation. Creating and initializing instances can be time-consuming.
5. **Use Inline Variables**: If possible, assign values to variables before evaluating the expression. This can help simplify the expression and reduce the number of times variables are looked up.
6. **Optimize Your Expressions**: Depending on the specific expressions we're evaluating, there might be opportunities for optimization. For example, replacing expensive operations with equivalent but faster operations.
7. **Use Compiled Code**: Some expression evaluation libraries allow us to compile expressions into native machine code for faster execution. The JclExprEval or future developing calculator based on such principles should support expression compilation.
8. **Evaluate in Bulk**: If we need to evaluate the same expression for multiple inputs, consider batching the evaluations together. This can take advantage of any optimizations that the library has for processing multiple inputs at once.

Considering the items above, the author of this article forked a new branch of the project called **unCalc (Universal Calculator)** and the project is under development. https://github.com/dima72/unCalc [14]

Conclusion. The advantages of using an arithmetic calculator as part of engineering calculation systems are more than obvious. Firstly, a dynamically interpreted mathematical expression can always be stored as a string, whether it is just a file or a database, as part of any program, and therefore be available for editing by the user, which in turn greatly facilitates the support and development of any software product. Now you do not need to recompile the entire project to fix an error in the formula. Secondly, the implementation of the apparatus of the mathematical calculator is compact, the reduction of the manual code for processing calculations, formulas always improving the quality of the product, allows you to separate the apparatus of calculations from the formulas themselves. Thirdly, knowledge of the construction of an arithmetic calculator helps to achieve a deeper understanding of the development of interpreters, compilers, and universal software. Considering current development trends in information technology, the language of the calculator implementation is of secondary importance. This calculator can be implemented as part of frameworks and languages that integrate with the cross-platform .NET framework, such as Oxygene and Hydra from RemObjects [15] or on the TMS XData platform [16]. The novelty of this approach lies in the provision of distributed engineering computing on the Cloud.

Список литературы

1. Библиотека кода JEDI распространяется на условиях публичной лицензии Mozilla (MPL). // https://github.com/project-jedi/jcl
2. Конструкция компилятора Wikipedians // PediaPress 189с.
3. Джон Левин flex & bison: Инструменты для обработки текста // O'Reilly Media 2009, 84с.
4. Оле-Йохан Даль, Эдсгер В. Дейкстра, Чарльз А. Р. Хоар Структурированное программирование // Acad. Пресса 1980
5. Джим Кеог, Дэвидсон Структуры данных: принципы и фундаментальные основы // Dreamtech Press 2004
6. Банников Н.А. Как писать переводчики. // http://www.stikriz.narod.ru/art/Interp.htm (дата посещения: 08/02/2023)
7. Ксавье Пачеко, Стив Тейшейра Руководство разработчика Borland Delphi 6 // Sams 2001
8. Марко Кэнти осваивает Borland Delphi // Wiley 2005, 913с.
9. Паллави Виджай Чаван, Ашиш Джадхав Теория автоматов и формальные языки // Elsevier Science 2023 112с.
10. Ханнес Хапке, Коул Ховард, Хобсон Лейн Обработка естественного языка в действии // Мэннинг 2019
11. Мануэль Рубио-Санчес Введение в рекурсивный Pr.
12. Björn Andrist, Viktor Sehr C++ High Performance Boost and Optimize the Performance of Your C++17 Code // Packt Publishing, 2018
13. Dan C. Marinescu Cloud Computing: Theory and Practice // Elsevier Science 2022
14. Dmitry Konnov unCalc // https://github.com/dima72/unCalc (date of visit: 08/14/2023)
15. RemObjects Software // https://www.remobjects.com (date of visit: 08/02/2023)

16. Delphi framework for multi-tier REST/JSON HTTP/HTTPS application server development and ORM remoting. // https://www.tmssoftware.com/site/xdata.asp (date of visit: 08/02/2023)

References

1. The JEDI Code Library is distributed under the terms of the Mozilla Public License (MPL). // https://github.com/project-jedi/jcl
2. By Wikipedians Compiler Construction // PediaPress p.189
3. John Levine flex & bison: Text Processing Tools // O'Reilly Media 2009, p. 84
4. Ole-Johan Dahl, Edsger W. Dijkstra, Charles A. R. Hoare Structured programming // Acad. Press 1980
5. Jim Keogh, Davidson Data Structures: Principles and Fundamentals // Dreamtech Press 2004
6. Bannikov N.A. How to write interpreters. // http://www.stikriz.narod.ru/art/Interp.htm (date of visit: 08/02/2023)
7. Xavier Pacheco, Steve Teixeira Borland Delphi 6 Developer's Guide // Sams 2001
8. Marco Canty Mastering Borland Delphi // Wiley 2005 p.913
9. Pallavi Vijay Chavan, Ashish Jadhav Automata Theory and Formal Languages // Elsevier Science 2023 p.112
10. Hannes Hapke, Cole Howard, Hobson Lane Natural Language Processing in Action // Manning 2019
11. Manuel Rubio-Sanchez Introduction to Recursive Programming // CRC Press 2017
12. Björn Andrist, Viktor Sehr C++ High Performance Boost and Optimize the Performance of Your C++17 Code // Packt Publishing, 2018
13. Dan C. Marinescu Cloud Computing: Theory and Practice // Elsevier Science 2022
14. Dmitry Konnov unCalc // https://github.com/dima72/unCalc (date of visit: 08/14/2023)
15. RemObjects Software // https://www.remobjects.com (date of visit: 08/02/2023)
16. Delphi framework for multi-tier REST/JSON HTTP/HTTPS application server development and ORM remoting. // https://www.tmssoftware.com/site/xdata.asp (date of visit: 08/02/2023)

Долгушева А.В., Таран В.В., Чернова С.В. Исследование системы защиты от вредоносных программ и кибератак // Международный журнал информационных технологий и энергоэффективности. – 2023. –
Т. 8 № 8(34) с. 17–23

Международный журнал информационных технологий и энергоэффективности

Сайт журнала:
http://www.openaccessscience.ru/index.php/ijcse/

УДК 004.056

ИССЛЕДОВАНИЕ СИСТЕМЫ ЗАЩИТЫ ОТ ВРЕДОНОСНЫХ ПРОГРАММ И КИБЕРАТАК

Долгушева А.В., [1]Таран В.В., Чернова С.В.
ФГБОУ ВО "ПОВОЛЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ И ИНФОРМАТИКИ", Самара, Россия (443010, Самарская область, город Самара,улица Льва Толстого, дом 23), e-mail: [1]kim.drive43@mail.ru

В данной статье рассматривается тема кибербезопасности и вирусов, какие угрозы существуют в интернете и какие меры безопасности необходимо принимать для защиты своих данных и личной информации. А также различные виды вирусов и их действия. Описывается, как вирусы попадают на компьютер, какие последствия они могут иметь и как защититься от них. Представлен список аскетов по исследование системы защиты от вредоносных программ и кибератак. В заключении подчеркивается важность защиты своих данных и личной информации в интернете, а также необходимость постоянного обновления мер безопасности для борьбы с угрозами кибербезопасности, распространенных во всем мире, и возможных способов их преодоления.

Ключевые слова: Кибератаки, вредоносные программы, система защиты.

INVESTIGATION OF THE SYSTEM OF PROTECTION AGAINST MALWARE AND CYBER ATTACKS

Dolgusheva A.V., [1]Taran V.V., Chernova S.V.
VOLGA STATE UNIVERSITY OF TELECOMMUNICATIONS AND INFORMATICS, Samara, Russia (443010, Samara region, Samara city, Leo Tolstoy street, 23), e-mail: [1]kim.drive43@mail.ru

This article discusses the topic of cybersecurity and viruses, what threats exist on the Internet and what security measures need to be taken to protect your data and personal information. As well as various types of viruses and their actions. It describes how viruses get on the computer, what consequences they can have and how to protect against them. The list of ascetics for the study of the system of protection against malware and cyber attacks is presented. The conclusion emphasizes the importance of protecting your data and personal information on the Internet, as well as the need for constant updating of security measures to combat cybersecurity threats common throughout the world, and possible ways to overcome them.

Keywords: Cyber attacks, malware, protection system.

Введение

Современный мир зависит от информационных технологий, и этот факт делает нас более уязвимыми к кибератакам и вирусам.

Кибератака — это попытка несанкционированного доступа к компьютерной системе, чтобы украсть, изменить или уничтожить данные. Кибератаки могут осуществляться с

помощью различных методов, таких как вирусы, троянские программы, фишинг и другие вредоносные действия.

Вирус — это программа, которая может копировать себя и распространяться на другие компьютеры. Вирусы могут привести к потере данных, нарушению работы компьютеров, краже личной информации и многим другим негативным последствиям. Вирусы могут распространяться через электронную почту, загрузки из Интернета, портативные устройства и другие способы.

Вирусной атакой, называется атака на удаленную/локальную компьютерную систему с использованием вредоносного программного обеспечения (вирусов). Они являются более изощренным методом доступа к секретной информации, поскольку хакеры используют специальные программы для работы на компьютере жертвы, а также для ее дальнейшего распространения (это вирусы и черви). Такие программы предназначены для поиска и передачи секретной информации своему владельцу или просто для повреждения системы безопасности и производительности компьютера жертвы. Принципы работы этих программ разные [1] .

Вредоносные программы и кибератаки

Вредоносные программы, также известные как малварь (malware), являются вредоносным программным обеспечением, разработанным для нанесения вреда компьютерным системам, сетям и пользователям. Они могут быть созданы злоумышленниками с различными целями, включая получение незаконного доступа к системе, кражу личных данных, финансовые мошенничества, шпионаж и прочие вредоносные действия.

Существует несколько типов вредоносных программ, включая:

Вирусы (Viruses): Это программы, которые могут размножаться и распространяться путем заражения других файлов или программ. Они прикрепляются к исполняемым файлам и могут внедряться в систему, повреждать файлы и распространяться на другие компьютеры через сети или носители информации.

Черви (Worms): Черви являются автономными программами, которые могут самостоятельно распространяться по компьютерным сетям. Они могут использовать уязвимости в сетевых протоколах или программном обеспечении для заражения компьютеров и их дальнейшего распространения [2].

Троянские программы (Trojans): Троянские программы скрываются под видом полезного или желаемого программного обеспечения, но при запуске выполняют вредоносные действия без ведома пользователя. Они могут открывать задние двери, собирать и передавать личные данные, создавать ботнеты и выполнять другие вредоносные функции.

Рекламное ПО (Adware): Рекламное программное обеспечение отображает навязчивую рекламу на компьютере пользователя. Оно может быть установлено с другими программами или распространяться через вредоносные сайты. Рекламное ПО может также собирать информацию о пользователе без его согласия.

Шпионское ПО (Spyware): Шпионское программное обеспечение отслеживает и собирает информацию о пользователе без его ведома и согласия. Оно может записывать

нажатия клавиш, отслеживать активность веб-браузера, собирать личные данные и передавать их злоумышленнику.

Кибератаки представляют собой злонамеренные действия, совершаемые в цифровом пространстве с целью нанести ущерб компьютерным системам, сетям или пользователям. Они могут быть осуществлены различными методами и иметь разные цели, включая кражу личных данных, финансовые мошенничества, нарушение работы систем, шпионаж и другие вредоносные действия.

Некоторые из распространенных типов кибератак включают:

Фишинг (Phishing): Это атака, при которой злоумышленник пытается обмануть пользователей, выдавая себя за надежное лицо или организацию. Целью фишинга является получение личных данных, таких как пароли, номера кредитных карт или банковские реквизиты.

Вредоносные программы (Malware): Как уже упоминалось ранее, вредоносные программы, такие как вирусы, черви, троянские программы и шпионское программное обеспечение, могут использоваться для целенаправленных атак на компьютерные системы, с целью нанести ущерб или получить несанкционированный доступ.

DDoS-атаки (Distributed Denial of Service): Это атаки, при которых злоумышленник пытается перегрузить целевую систему или сеть большим количеством запросов, что приводит к отказу в обслуживании для легитимных пользователей.

Взлом (Hacking): Взлом может быть направлен на получение несанкционированного доступа к компьютерной системе или сети, с целью кражи данных, изменения настроек или выполнения других вредоносных действий.

Социальная инженерия (Social Engineering): Это метод манипулирования людьми, чтобы получить доступ к конфиденциальной информации или выполнить действия, которые могут нанести ущерб. Примерами могут быть обман пользователей, чтобы получить их пароли, или убеждение сотрудников предоставить доступ к системе [3].

Способы защиты от вредоносных программ и кибератак

Существует несколько способов защиты от вредоносных программ:

1. Установка антивирусного программного обеспечения: Используйте надежное антивирусное программное обеспечение и регулярно обновляйте его. Антивирусное ПО поможет обнаружить и удалить вредоносные программы с вашего компьютера.

2. Обновление операционной системы и программ: Регулярно обновляйте операционную систему и все установленные программы. Обновления часто содержат исправления уязвимостей, которые могут быть использованы злоумышленниками для атак.

3. Осторожность при скачивании и установке программ: Загружайте программы только с надежных и официальных источников. Будьте внимательны при установке программ и не разрешайте им получать необходимые разрешения, если вы не уверены в их надежности.

4. Осмотрительность в интернете: Будьте осторожны при открытии вложений в электронных письмах, переходе по подозрительным ссылкам или скачивании файлов из ненадежных источников. Вредоносные программы могут быть скрыты в этих элементах.

5. Включение брандмауэра: Убедитесь, что брандмауэр на вашем компьютере включен. Брандмауэр помогает контролировать входящий и исходящий сетевой трафик и блокировать подозрительные соединения.

6. Резервное копирование данных: Регулярно создавайте резервные копии важных данных. В случае атаки или заражения вредоносной программой, резервные копии помогут восстановить информацию.

7. Обновление браузера и использование безопасного соединения: Обновляйте ваш браузер до последней версии и предпочитайте использование безопасного HTTPS-соединения при посещении веб-сайтов.

8. Обучение пользователей: Проводите обучение пользователей по базовым правилам безопасности, таким как неоткрывание подозрительных ссылок, нескачивание файлов из ненадежных источников и осмотрительность при взаимодействии с электронной почтой.

9. Многофакторная аутентификация: Включите многофакторную аутентификацию для важных аккаунтов. Это дополнительный слой защиты, требующий не только пароль, но и дополнительный фактор, такой как одноразовый код или отпечаток пальца.

10. Мониторинг и обнаружение угроз: Используйте программное обеспечение для мониторинга и обнаружения угроз, которое поможет выявить аномальную активность и своевременно предупредить о возможных атаках.

Все эти меры в комбинации могут помочь защитить вашу систему от вредоносных программ и повысить уровень безопасности [4-6].

Для защиты от кибератак существует ряд мер и методов. Вот несколько основных способов защиты:

1. Постоянное обновление программного обеспечения: Регулярно обновляйте операционную систему, приложения и программное обеспечение до последних версий. Обновления часто содержат исправления уязвимостей, которые могут быть использованы злоумышленниками для проведения атак.

2. Использование надежного антивирусного программного обеспечения: Установите и регулярно обновляйте антивирусное программное обеспечение на своем компьютере. Это поможет обнаруживать и блокировать вредоносные программы, включая программы-шпионы и троянские кони.

3. Файервол: Установите и настройте брандмауэр на своем компьютере или сетевом устройстве. Файервол поможет контролировать входящий и исходящий сетевой трафик и блокировать подозрительные соединения.

4. Сильные пароли и многофакторная аутентификация: Используйте сложные и уникальные пароли для своих онлайн-аккаунтов. Рекомендуется также включить многофакторную аутентификацию, чтобы требовать дополнительный фактор подтверждения, такой как одноразовый код, при входе в аккаунт.

5. Осмотрительность в интернете: Будьте внимательны при посещении веб-сайтов и открытии ссылок или вложений из ненадежных источников. Избегайте скачивания файлов с подозрительных сайтов и не предоставляйте личные данные на ненадежных веб-ресурсах.

6. Обучение пользователей: Обучите себя и других пользователям основным правилам безопасности в сети. Это может включать осведомленность о методах фишинга, социальной

инженерии и других видов кибератак, а также обучение о том, как реагировать и защищаться от них.

7. Регулярное резервное копирование данных: Регулярно создавайте резервные копии важных данных и храните их в надежном месте. Это поможет восстановить данные в случае утраты или шифрования в результате кибератаки.

8. Ограничение привилегий доступа: Ограничьте привилегии доступа пользователей и приложений на своем компьютере или сервере. Убедитесь, что только необходимые пользователи имеют доступ к конфиденциальной информации или системным ресурсам.

9. Мониторинг и обнаружение аномальной активности: Используйте специальное программное обеспечение для мониторинга и обнаружения аномалий в сети. Это поможет выявить подозрительную активность и своевременно предупредить о возможных кибератаках.

10. Регулярные аудиты и тестирование на проникновение: Проводите регулярные аудиты безопасности своей системы, а также тестирование на проникновение, чтобы выявить уязвимости и устранить их до того, как они могут быть использованы злоумышленниками.

Это лишь некоторые из основных способов защиты от кибератак. Важно постоянно быть внимательными и осведомленными о новых угрозах и методах защиты для эффективной борьбы с киберугрозами.

Исследование системы защиты от вредоносных программ и кибератак

Исследование системы защиты от вредоносных программ и кибератак является важным аспектом обеспечения информационной безопасности. Вот некоторые ключевые аспекты, которые могут быть включены в такое исследование:

- Угрозный анализ: Изучение существующих угроз и вредоносных программ, анализ их характеристик, методов распространения и воздействия на системы. Это позволит определить наиболее вероятные угрозы для вашей системы и разработать соответствующие контрмеры.
- Оценка уязвимостей: Исследование и анализ уязвимостей в вашей системе, которые могут быть использованы злоумышленниками для вторжения или проведения кибератак. Это включает оценку слабых мест в сетевой инфраструктуре, программном обеспечении, конфигурации систем и процедур безопасности.
- Анализ архитектуры защиты: Изучение текущей архитектуры защиты вашей системы, включая физические, сетевые и программные механизмы защиты. Оценка эффективности существующих мер защиты и их соответствие современным стандартам безопасности.
- Планирование и реализация мер безопасности: Разработка и реализация мер безопасности, направленных на защиту системы от вредоносных программ и кибератак. Это может включать обновление программного обеспечения, установку брандмауэров, антивирусных программ и других механизмов защиты, настройку систем мониторинга и регистрации событий, резервное копирование данных и регулярное обновление политик безопасности.
- Обучение и осведомленность пользователей: Разработка программ обучения пользователей по вопросам информационной безопасности, чтобы повысить их

осведомленность о потенциальных угрозах и о том, как следовать политикам и процедурам безопасности. Это может включать проведение тренингов, распространение информационных материалов и организацию регулярных проверок осведомленности.

- Мониторинг и обнаружение инцидентов: Разработка механизмов мониторинга и обнаружения инцидентов, которые позволят оперативно обнаружить и реагировать на потенциальные атаки и нарушения безопасности. Это включает настройку систем мониторинга сетевого трафика, анализ журналов событий, использование инструментов обнаружения вторжений и других методов обнаружения аномалий.
- Реагирование на инциденты: Разработка планов реагирования на инциденты и проведение учений по их исполнению. Это включает разработку процедур для обработки инцидентов, механизмов реагирования, координации сотрудников и взаимодействия с внешними экспертами по безопасности.

Важно отметить, что эти аспекты зависят от конкретных требований и особенностей вашей системы и должны быть адаптированы под ваши нужды. Рекомендуется также обратиться к специалистам в области информационной безопасности для получения более детальных рекомендаций и консультаций.

Заключение

Современный мир все больше зависит от информационных технологий, что делает нас более уязвимыми к кибератакам и вирусам. Поэтому понимание того, как работают вирусы и вредоносные программы, а также методы защиты от них, является крайне важным для обеспечения информационной безопасности и защиты личных данных.

В целом, защита от вирусов и вредоносных программ является актуальной проблемой в современном мире. Несмотря на то, что существует множество программ и методов, которые обеспечивают защиту, вирусы все равно находят способы проникновения в наши компьютеры и устройства. Важно быть внимательными и следить за своей информационной безопасностью, а также постоянно обновлять программное обеспечение и использовать антивирусные программы, чтобы минимизировать риски.

Список литературы

1. Вирусы и другие вредоносные программы // Национальный Открытый Университет-2017—URL: https://intuit.ru/studies/courses/76/76/lecture/27946(дата обращения: 26.06.2023)
2. 13 различных типов вредоносных программ // New-Science.ru—2021—URL: https://new-science.ru/13-razlichnyh-tipov-vredonosnyh-programm/(дата обращения: 26.06.2023)
3. Палаева Л. В. Основные виды кибератак на автоматизированные системы управления технологическим процессом и средства защиты от них / Л. В. Палаева, А. М. Хафизов, А. М. Гилязетдинова // Фундаментальные исследования.—2017.—10—С.507–511.
4. Алеексеев П. Антивирусы. Настраиваем защиту компьютера от вирусов / П. Алеексеев, Д. Козлов, Р. Прокди—Москва: Наука и Техника, 2018.—915 с.
5. Шаньгин В.Ф. Защита компьютерной информации / В.Ф. Шаньгин—Москва: ДМК Пресс, 2020.—544 с.

6. Александров К.П. Компьютер без сбоев, вирусов и проблем / К.П. Александров, Р.Г. Прокди—Москва: Наука и техника, 2017.—192 с.

References

1. Viruses and other malicious programs // National Open University. — 2017 — URL: https://intuit.ru/studies/courses/76/76/lecture/27946 (accessed: 06/26/2023)
2. 13 different types of malware // New-Science.ru . — 2021 — URL: https://new-science.ru/13-razlichnyh-tipov-vredonosnyh-programm / (accessed: 06/26/2023)
3. Palaeva L. V. NEW TYPES OF CYBERATTACKS ON AUTOMATED PROCESS CONTROL SYSTEMS AND MEANS OF PROTECTION AGAINST THEM / L. V. Palaeva, A.M. Hafizov, A.M. Gilyazetdinova // Fundamental research. — 2017. — 10. — pp. 507-511.
4. Aleekseev P. Antiviruses. Setting up computer protection against viruses / P. Aleekseev, D. Kozlov, R. Prokdi—Moscow: Science and Technology, 2018.—p.915
5. Shangin V.F. Protection of computer information / V.F. Shangin — Moscow: DMK Press, 2020.—p.544
6. Alexandrov K.P. Computer without failures, viruses and problems/K.P. Alexandrov, R.G. Prokdi—Moscow: Science and Technology, 2017.—p.192

Сафин М.А., Сафиуллина А.Ф. Автоматизация процессов переработки и рефининга нефти // Международный журнал информационных технологий и энергоэффективности. – 2023. – Т. 8 № 8(34) с. 24–27

Международный журнал информационных технологий и энергоэффективности

Сайт журнала:
http://www.openaccessscience.ru/index.php/ijcse/

УДК 681.5

АВТОМАТИЗАЦИЯ ПРОЦЕССОВ ПЕРЕРАБОТКИ И РЕФИНИНГА НЕФТИ

Сафин М.А., [1]Сафиуллина А.Ф.
ФГБОУ ВО "КАЗАНСКИЙ ГОСУДАРСТВЕННЫЙ ЭНЕРГЕТИЧЕСКИЙ УНИВЕРСИТЕТ", Казань Россия (420066, Республика Татарстан, город Казань, Красносельская ул, д. 51), *e-mail: [1]alsukizikay@gmail.com*

В данной статье рассматриваются современные аспекты автоматизации процессов переработки и рефининга нефти. Выявляется её роль и преимущества, а также анализируются основные вызовы и перспективы развития в этой области.

Ключевые слова: Рефининг, переработка нефти, автоматизация, нефтегазовая отрасль, устойчивое развитие.

AUTOMATION OF OIL REFINING AND REFINING PROCESSES

Safin M.A., [1]**Safiullina A.F.**
KAZAN STATE POWER ENGINEERING UNIVERSITY, Kazan, Russia (420066, Republik of Tatarstan, Kazan city, Krasnoselskaya street, 51), e-mail: [1]alsukizikay@gmail.com

This article discusses modern aspects of automation of oil refining and refining processes. Its role and advantages are revealed, as well as the main challenges and prospects for development in this area are analyzed.

Keywords: Refining, oil refining, automation, oil and gas industry, sustainable development.

В современном мире нефтегазовая промышленность играет важнейшую роль, обеспечивая энергетическую безопасность и поддерживая функционирование глобальной экономики. В этом контексте переработка и рефининг нефти выступают как критические этапы, преобразующие сырой нефтепродукт в ценные и разнообразные энергетические и химические компоненты. От топлива для транспорта до пластмасс и химических веществ, продукты рефининга играют важную роль в повседневной жизни.

С целью повышения эффективности, улучшения качества продукции и сокращения негативного воздействия на окружающую среду нефтегазовая промышленность стремится к постоянному совершенствованию своих процессов. И в этом контексте, автоматизация выходит на первый план. Автоматизация процессов переработки и рефининга нефти стала ключевой стратегией, обеспечивающей эффективность, точность и безопасность в производственных операциях.

Основной целью автоматизации в данных процессах является достижение оптимальной производительности, минимизация рисков и обеспечение стабильности операций. Автоматизация позволяет не только улучшить качество продукции и снизить операционные издержки, но и сократить воздействие человеческого фактора на производство, что особенно

актуально в условиях сложных и потенциально опасных рабочих сред. В данной статье мы подробно рассмотрим современные аспекты автоматизации процессов переработки и рефининга нефти, выявим её роль и преимущества, а также рассмотрим основные вызовы и перспективы развития в этой области.

Процессы переработки и рефининга нефти представляют собой сложную и многоэтапную последовательность операций, направленных на превращение сырой нефти в ценные продукты с высокой добавленной стоимостью. Обзор ключевых этапов переработки и рефининга [1, 2] отражает многообразие технологий и методов, используемых в промышленности для получения широкого спектра продуктов – от топлива до химических компонентов.

Процесс начинается с дистилляции, где нефтепродукты разделяются на фракции с различными температурными диапазонами. Следующим этапом является крекинг, включая каталитический крекинг, который обеспечивает разделение более тяжёлых углеводородов на более лёгкие. После этого идут этапы гидроочистки, гидрокрекинга, алькирования и другие, каждый из которых направлен на улучшение качества продуктов и повышение их соответствия требованиям рынка [3].

Сложности и вызовы, связанные с каждым этапом, нередко обусловлены разнообразными химическими и физическими свойствами сырой нефти. Эти вызовы могут включать в себя необходимость обеспечения оптимальных температур и давлений для реакций, управление катализаторами, предотвращение образования отложений и коррозии, а также соблюдение строгих нормативов по окружающей среде.

Современная нефтегазовая промышленность активно внедряет автоматизацию на различных этапах процессов переработки и рефининга [4]. На этапе дистилляции и фракционирования используются автоматические системы контроля и регулирования, обеспечивающие оптимальные параметры процессов. Для крекинга применяются автоматические каталитические реакторы [5], способные поддерживать стабильные условия и обеспечивать более высокую эффективность.

Примером автоматизированных устройств может служить система мониторинга и контроля параметров гидроочистки, обеспечивающая точность регулирования реакционных условий [6]. На этапах алькирования и изомеризации также широко используются автоматизированные системы для точного дозирования катализаторов и реагентов [7].

Автоматизация процессов переработки и рефининга предоставляет значительные преимущества. Повышение эффективности и точности регулирования позволяет достигать более высокой производительности и качества продукции. Анализ данных и использование алгоритмов искусственного интеллекта способствуют более точной оптимизации процессов и снижению ошибок [8].

Улучшение контроля над производством и оперативное реагирование на изменения параметров обеспечивают минимизацию рисков, связанных с возможными аварийными ситуациями. Безопасность персонала и окружающей среды также усиливается благодаря уменьшению человеческого вмешательства в опасные операции.

Тем не менее, внедрение автоматизации также сопряжено с определенными вызовами. Возможные технические сбои и проблемы с кибербезопасностью могут привести к остановке

производства и потенциальным угрозам данных [9]. Кроме того, автоматизация может вызвать потерю рабочих мест и снижение спроса на квалифицированных специалистов.

Важно подчеркнуть, что баланс между автоматизацией и человеческим участием необходим для обеспечения гибкости, адаптивности и надежности производства. Человеческий опыт и интуиция остаются важными факторами при принятии решений в сложных ситуациях [10].

Современные тенденции в нефтегазовой промышленности указывают на непрерывное развитие и инновации в области автоматизации процессов переработки и рефининга нефти. Проекция технологических трендов подразумевает дальнейшее усиление интеграции автоматизации на всех этапах производства [11].

В будущем, автоматизация может оказать значительное влияние на производственные процессы и результаты. Переход к полностью автономным системам контроля, оптимизации и управления может повысить эффективность, устранить операторские ошибки и минимизировать риски, связанные с человеческим фактором [12]. Применение аналитики больших данных и искусственного интеллекта позволит точнее предсказывать и управлять процессами, учитывая переменные внешние факторы.

Автоматизация становится ключевым фактором для переработки и рефининга нефти в будущем. Важно понимать, что эти изменения не только повысят эффективность и экономическую состоятельность, но и приведут к снижению негативного влияния на окружающую среду [13]. Перспективы автоматизации огромны, и она может стать одним из ключевых инструментов в достижении целей устойчивого развития отрасли.

Заключительные мысли о необходимости продолжения исследований и инвестиций в данной сфере подчеркивают важность не только внедрения, но и постоянного совершенствования автоматизированных систем и технологий. Осознание значимости баланса между технологическими инновациями и человеческим мастерством позволит обеспечить устойчивое развитие нефтегазовой промышленности в будущем

Список литературы

1. Петров, А. Б. Промышленная рефинерия: технология и автоматизация. Лань, 2018.
2. Parkash, S. Refining Processes Handbook. Gulf Professional Publishing, 2003.
3. Семенов, А. Л. Основы нефтегазовой индустрии: технология и оборудование. Горячая линия - Телеком, 2012.
4. Иванов, И. И. Автоматизация и управление в нефтегазовой промышленности. Нефть и газ, 2015.
5. Trivedi, B. L. Automation of Refinery Processes. Wiley, 2013.
6. Bonvin, J. M. Process Automation Handbook: A Guide to Theory and Practice. Springer, 2010.
7. Francis, R. A. Automation in Petroleum Refining and Petrochemical Industries. CRC Press, 1995.
8. Lamb, F. Industrial Automation: Hands-On. McGraw-Hill Education, 2013.
9. Macaulay, T., Singer, B. L. Cybersecurity for Industrial Control Systems: SCADA, DCS, PLC, HMI, and SIS. Auerbach Publications, 2011.
10. Kaplan, J. Humans Need Not Apply: A Guide to Wealth and Work in the Age of Artificial Intelligence. Yale University Press, 2015.

11. Гупта, М. "Будущие тенденции в автоматизации процессов переработки нефти." В: Будущее автоматизации в нефтегазовой промышленности. Издательство CRC Press, 2018.
12. Сараванан, К. "Искусственный интеллект в рефининге." В: Инновации в автоматизации процессов переработки и рефининга нефти. Издательство CRC Press, 2020.
13. Тузинский, А., Абдулсада, Х. "Устойчивая переработка и рефининг." В: Современные подходы к автоматизации и оптимизации нефтегазовой промышленности. Издательство CRC Press, 2021.

References

1. Petrov, A. B. Industrial refineries: technology and automation. Lan, 2018.
2. Parkash, S. Handbook of processing processes. Gulf Professional Publishing, 2003.
3. Semenov, A. L. Fundamentals of the oil and gas industry: technology and equipment. Hotline - Telecom, 2012.
4. Ivanov, I. I. Automation and control in the oil and gas industry. Oil and Gas, 2015.
5. Trivedi B. L. Automation of oil refining processes. Wiley, 2013.
6. Bonvin, J. M. Handbook of Process Automation: A Guide to Theory and Practice. Springer, 2010.
7. Francis, R. A. Automation in the oil refining and petrochemical industry. CRC Press, 1995.
8. Lamb, F. Industrial Automation: Practical Experience. McGraw-Hill Education, 2013.
9. Macaulay T., Singer B. L. Cybersecurity of industrial control systems: SCADA, DCS, PLC, HMI and SIS. Auerbach Publications, 2011.
10. Kaplan, J. People Don't Need to Apply: A Guide to Wealth and Work in the Age of Artificial
11. Gupta, M. "Future Trends in Automation of Oil Refining Processes." Q: The Future of Automation in the Oil and Gas Industry. CRC Press, 2018.
12. Saravanan, K. "Artificial Intelligence in Refining." Q: Innovations in the automation of oil refining and refining processes. CRC Press, 2020.
13. Tuzinsky, A., Abdulsada, H. "Sustainable processing and refining." In: Modern approaches to automation and optimization of the oil and gas industry. CRC Press, 2021.

Павлов А.В., Хрусталева М.С. Энергосбережение в центрах обработки данных // Международный журнал информационных технологий и энергоэффективности. – 2023. – Т. 8 № 8(34) с. 28–31

Международный журнал информационных технологий и энергоэффективности

Сайт журнала:
http://www.openaccessscience.ru/index.php/ijcse/

УДК 004.

ЭНЕРГОСБЕРЕЖЕНИЕ В ЦЕНТРАХ ОБРАБОТКИ ДАННЫХ

[1]Павлов А.В., Хрусталева М.С.
ФГБОУ ВО "ТВЕРСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ", Тверь, Россия (170026, Тверская область, город Тверь, наб. Афанасия Никитина, д.22), e-mail: [1] sokolhawk98@gmail.com

В статье обсуждаются проблемы и возможности управления центром обработки данных как узлом интеллектуальной сети. Связь между сетью и центром обработки данных принимает форму изменяющейся во времени и зависящей от потребляемой мощности цены на электроэнергию. Рассмотрена модель, учитывающая как вычислительные, так и физические характеристики центра обработки данных, а также их взаимодействие. Обсуждаются два подхода к управлению. Первый подход к управлению не учитывает взаимодействие между вычислительными и тепловыми характеристиками ЦОД. Второй регулятор учитывает связь между расчетными и тепловыми характеристиками.

Ключевые слова: Центр обработки данных, энергосистема, энергопотребление, экономия электроэнергии, эффективность использования электроэнергии, стоимость электроэнергии.

ENERGY SAVING IN DATA CENTERS

[1]Pavlov A.V., Khrustaleva M.S.
TVER STATE TECHNICAL UNIVERSITY, Tver, Russia (170026, Tver region, Tver city, Afanasiya Nikitin embankment, 22), e-mail: [1]sokolhawk98@gmail.com

The article discusses the problems and possibilities of managing a data center as a node of an intelligent network. The connection between the network and the data center takes the form of a time-varying, power-dependent price of electricity. A model is considered that takes into account both the computational and physical characteristics of the data center, as well as their interaction. Two approaches to control are discussed. The first approach to control does not take into account the interaction between the computing and thermal characteristics of the data center. The second controller takes into account the relationship between design and thermal characteristics.

Keywords: Data center, power system, energy consumption, energy savings, energy efficiency, electricity cost.

Количество центров обработки данных значительно увеличилось во всем мире, чему способствует растущий спрос на услуги удаленного хранения и облачных вычислений. Энергопотребление центров обработки данных также значительно увеличилось. Эффективное питание и охлаждение центров обработки данных также стало серьезной проблемой. Современные центры обработки данных (ЦОД) потребляют около 1–2% электричества, вырабатываемого в мире [1]. Энергия, потребляемая для вычислений и охлаждения, является доминирующей в управлении временем работы центра обработки данных и эксплуатационных расходах, и даже небольшой процент снижения энергопотребления может иметь большое влияние на экономику эксплуатации центра обработки данных.

Мерой эффективности центра обработки данных, обычно используемой в промышленности, является эффективность использования энергии (ЭИЭ) [2]. ЭИЭ центра обработки данных определяется как отношение между общей потребляемой мощностью центра обработки данных и потребляемой мощностью информационных технологий. Значение ЭИЭ, равное 1,0, указывает на то, что вся потребляемая мощность центра обработки данных приходится на ИТ. Значения ЭИЭ, собранные более чем в 60 различных центрах обработки данных, обсуждаются в [2]. Среднее значение ЭИЭ составляет 2,03, восемь центров обработки данных имеют значения ЭИЭ ниже 1,5, а шесть центров обработки данных имеют значения ЭИЭ выше 2,75. Большинство центров обработки данных имеют значения ЭИЭ от 1,5 до 2 [2]. Одним из недостатков индекса ЭИЭ является то, что он не показывает, насколько эффективно используются информационные технологии.

С точки зрения энергосистемы особенностями центров обработки данных являются их высокое значение энергопотребления на квадратный метр и масштаб времени, в котором энергопотребление можно контролировать, например, в минутном масштабе. На нерегулируемом рынке электроэнергии цена на электроэнергию может меняться со временем, а также в зависимости от географического положения. Жизнеспособным методом снижения изменчивости стоимости электроэнергии является требование к некоторым потребителям ограничивать потребление электроэнергии по запросу от сети. Такой метод, уже применяемый некоторыми независимыми системными операторами, называется программой реагирования на спрос.

Мы рассматриваем случай, описанный в источнике [3], когда контроллер центра обработки данных может использовать два соглашения об уровне обслуживания. Первое соглашение регулирует доход центра обработки данных на основе качества обслуживания, предоставляемого пользователям. Второе соглашение регулирует стоимость электроэнергии. Рассмотрим случай, когда центр обработки данных участвует в программе реагирования на спрос с электросетью. Связь между электросетью и центром обработки данных принимает форму изменяющейся во времени и зависящей от потребляемой мощности цены на электроэнергию. Пока среднее энергопотребление центра обработки данных поддерживается ниже порогового значения, изменяющегося во времени, центр обработки данных покупает электроэнергию по сниженной цене. Когда среднее энергопотребление центра обработки данных превышает изменяющийся во времени порог, дополнительная энергия предоставляется по более высокой цене. Средние значения энергопотребления центра обработки данных рассчитываются за заданное временное окно.

Технологии центров обработки данных можно условно разделить на три группы: информационные технологии, системы охлаждения и технологии поддержки [3,4]. Информационные технологии включают в себя серверы, устройства хранения и компоненты, связанные с сетью, такие как коммутаторы, брандмауэры и маршрутизаторы. Системы охлаждения включают в себя такие компоненты, как кондиционеры машинного зала и вентиляторы. К классу вспомогательных технологий относятся такие устройства, как батареи, генераторы резервного питания, источники бесперебойного питания, блоки распределения питания и т. д.

Центры обработки данных представляют собой крупномасштабные системы с масштабами времени от миллисекунд до десятков минут. Исполнительные механизмы,

которые могут использоваться алгоритмами управления, неоднородны по своей природе, и их эффекты актуальны в различных временных и пространственных масштабах. Например, методы динамического масштабирования напряжения и частоты работают на миллисекундном уровне и влияют на энергопотребление отдельного сервера [5]. В масштабе стойки миграция виртуальных машин влияет на энергопотребление нескольких серверов и работает в минутном масштабе [6]. В таком сценарии иерархическая архитектура управления является желательным подходом к управлению. Как и в работе [3], мы рассматриваем иерархически-распределенный подход к управлению ЦОД. На самом высоком уровне иерархии мы рассматриваем централизованный контроллер, называемый контроллером центра обработки данных. Контроллер центра обработки данных предоставляет набор границ и оптимальных уставок для контроллеров более низкого уровня, которые затем работают независимо друг от друга [4]. Отдельные стойки или наборы стоек моделируются как отдельные ИТ-компоненты, называемые зонами. В этой статье рассматривается самый высокий уровень иерархии и связанная сетевая модель. Первая сеть называется вычислительной сетью и определяет взаимосвязь между выполнением рабочей нагрузки, качеством обслуживания и количеством энергии, потребляемой каждой зоной. Вторая сеть называется тепловой сетью и определяет теплообмен между устройствами в центре обработки данных. На Рисунке 1 представлено графическое представление предлагаемой модели.

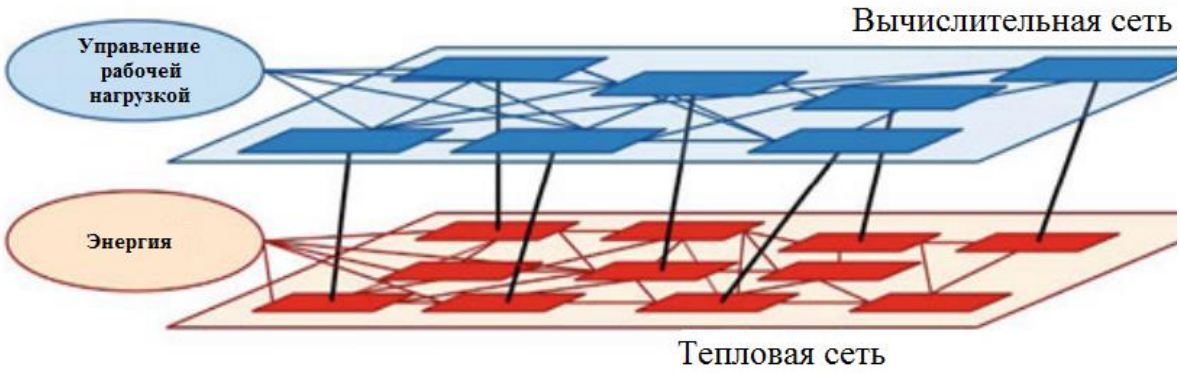


Рисунок 1 – Сетевая модель центра обработки данных

Зоны имеют как вычислительные, так и физические характеристики. Эти характеристики представлены двумя связанными узлами. Один узел моделирует вычислительные характеристики зоны, а другой узел моделирует физические характеристики зоны. Первый узел относится к вычислительной сети, а второй — к тепловой сети. Для упрощения записи будем считать, что два связанных узла, представляющих зону, имеют одинаковый индекс в соответствующих сетях, т. е. *i*-й узел вычислительной сети связан с *i*-м узлом тепловой сети. Среди устройств в составе системы охлаждения мы остановимся на блоках кондиционеров машинного зала. Другими устройствами, такими как вентиляторы, можно управлять на более низких уровнях иерархии. Блок кондиционеров машинного зала имеет только физические характеристики, которые представлены одним узлом тепловой сети.

Для снижения потребления электроэнергии необходимо добиться энергосбережения на каждом уровне иерархической системы. Только это приведет к ощутимой экономии не только в ЦОД, но и в стране, обеспечивающей его электроэнергией.

Список литературы

1. Дата-центр без потерь электроэнергии. Cnews [Электронный ресурс]. URL: https://translated.turbopages.org/proxy_u/en-ru.ru.77412d65-64dc9bc5-11cd7738-74722d776562/https/en.wikipedia.org/wiki/List_of_major_power_outages (дата обращения: 010.08.2023).
2. The Green Grid: The green grid data center power efficiency metrics: PUE and DCiE. White paper, Technical Committee. 2007
3. Parolini L., Sinopoli B., Krogh B.H. Model predictive control of data centers in the smart grid scenario // 18th International Federation of Automatic Control (IFAC) World Congress (Milano, Italy). 2011. Vol 18, Part 1.
4. Parolini L., Tolia N., Sinopoli B., Krogh B.H. A cyber-physical systems approach to energy management in data centers // First international conference on cyber-physical systems (Stockholm, Sweden). 2010. pp 168–177.
5. Aydin H., Zhu D. Reliability-aware energy management for periodic real-time tasks // IEEE Trans Comput. 2009. № 58(10). pp 1382–1397.
6. Jin H., Deng L., Wu S., Shi X., Pan X. Live virtual machine migration with adaptive, memory compression // In: Proc. IEEE Int. Conf. Cluster Computing and Workshops CLUSTER. 2009. pp 1–10/

References

1. Data center without power loss. Cnews [Online]. URL: https://translated.turbopages.org/proxy_u/en-ru.ru.77412d65-64dc9bc5-11cd7738-74722d776562/https/en.wikipedia.org/wiki/List_of_major_power_outages (дата обращения: 010.08.2023).
2. The Green Grid: The green grid data center power efficiency metrics: PUE and DCiE. White paper, Technical Committee. 2007
3. Parolini L., Sinopoli B., Krogh B.H. Model predictive control of data centers in the smart grid scenario // 18th International Federation of Automatic Control (IFAC) World Congress (Milano, Italy). 2011. Vol 18, Part 1.
4. Parolini L., Tolia N., Sinopoli B., Krogh B.H. A cyber-physical systems approach to energy management in data centers // First international conference on cyber-physical systems (Stockholm, Sweden). 2010. pp 168–177.
5. Aydin H., Zhu D. Reliability-aware energy management for periodic real-time tasks // IEEE Trans Comput. 2009. № 58(10). pp 1382–1397.
6. Jin H., Deng L., Wu S., Shi X., Pan X. Live virtual machine migration with adaptive, memory compression // In: Proc. IEEE Int. Conf. Cluster Computing and Workshops CLUSTER. 2009. pp 1–10/

Сафин М.А., Сафиуллина А.Ф. А. Безопасность и автоматизация в условиях нефтегазовой промышленности // Международный журнал информационных технологий и энергоэффективности. – 2023. –
Т. 8 № 8(34) с. 32–36

Международный журнал информационных технологий и энергоэффективности

Сайт журнала:
http://www.openaccessscience.ru/index.php/ijcse/

УДК 004.9

БЕЗОПАСНОСТЬ И АВТОМАТИЗАЦИЯ В УСЛОВИЯХ НЕФТЕГАЗОВОЙ ПРОМЫШЛЕННОСТИ

Сафин М.А., [1]Сафиуллина А.Ф.
ФГБОУ ВО "КАЗАНСКИЙ ГОСУДАРСТВЕННЫЙ ЭНЕРГЕТИЧЕСКИЙ УНИВЕРСИТЕТ", Казань Россия (420066, Республика Татарстан, город Казань, Красносельская ул, д. 51), *e-mail: [1]alsukizikay@gmail.com*

Статья рассматривает применения автоматизированных систем в различных сферах нефтегазовой индустрии для обеспечения безопасности персонала и окружающей среды при максимальной эффективности. Выявляются недостатки и преимущества инновационных технологий, используемых для минимизации рисков и потерь в промышленных процессах.

Ключевые слова: Нефтегазовая отрасль, автоматизация, безопасность, минимизация рисков, инновации, устойчивое развитие.

SAFETY AND AUTOMATION IN THE OIL AND GAS INDUSTRY

Safin M.A., [1]**Safiullina A.F.**
KAZAN STATE POWER ENGINEERING UNIVERSITY, Kazan, Russia (420066, Republik of Tatarstan, Kazan city, Krasnoselskaya street, 51), e-mail: [1]alsukizikay@gmail.com

The article examines the use of automated systems in various areas of the oil and gas industry to ensure the safety of personnel and the environment with maximum efficiency. The disadvantages and advantages of innovative technologies used to minimize risks and losses in industrial processes are revealed.

Keywords: Oil and gas industry, automation, safety, risk minimization, innovation, sustainable development.

Нефтегазовая промышленность, являясь одной из ключевых отраслей мировой экономики, обеспечивает снабжение энергоресурсами, необходимыми для различных сфер жизни. Однако этот важный сектор также сопряжен с рядом серьезных вызовов, включая сложные технические процессы, опасности для окружающей среды и человеческого здоровья. В контексте стремительно развивающейся технологической эпохи автоматизация играет решающую роль в обеспечении безопасности и эффективности в нефтегазовой отрасли.

Эта статья посвящена глубокому анализу взаимосвязи между безопасностью и автоматизацией в нефтегазовой промышленности. Современные технологические решения и инновации переворачивают привычные представления о производственных процессах, предоставляя возможности повышения эффективности и минимизации рисков. В данной статье мы рассмотрим ключевые аспекты влияния автоматизации на безопасность в нефтегазовой отрасли, а также проанализируем вызовы и риски, которые могут сопровождать этот путь.

Процессы автоматизации и безопасности в нефтегазовой промышленности представляют сложное взаимодействие между технологическими достижениями и человеческим опытом. С одной стороны, автоматизация способствует исключению ошибок и минимизации рисков, а с другой – она требует тщательного управления и контроля, чтобы не ослабить роль человека в обеспечении безопасности.

В современной нефтегазовой промышленности, где сложные технологические процессы и высокие стандарты безопасности играют решающую роль, автоматизация становится ключевым инструментом для обеспечения безопасности операций. В данной статье мы рассмотрим, как автоматизация влияет на обеспечение безопасности в этой отрасли, а также выявим вызовы и риски, связанные с этим процессом.

Автоматизация в нефтегазовой промышленности представляет собой системный подход к оптимизации операций, сокращению человеческого вмешательства и снижению рисков человеческих ошибок. Путем применения автоматизированных систем в различных сферах, от бурения до переработки, отрасль стремится к повышению эффективности и одновременно к обеспечению безопасности персонала и окружающей среды. Примером такой взаимосвязи может служить внедрение автоматизированных систем контроля и реагирования на предупреждения, что позволяет оперативно предотвращать аварии и минимизировать их последствия [1].

Технологические решения для обеспечения безопасности в условиях автоматизации включают широкий спектр инновационных методов. Сенсорные системы и системы мониторинга играют важную роль в раннем выявлении потенциальных аварийных ситуаций, обеспечивая возможность быстрого реагирования и минимизации угроз [2]. Автоматизированные системы пожаротушения и аварийного отключения оборудования гарантируют оперативное вмешательство даже в условиях экстремальных событий, снижая риски для персонала и окружающей среды [3]. Применение искусственного интеллекта и аналитики данных в предсказании потенциальных угроз безопасности позволяет создать превентивные меры, направленные на исключение аварийных ситуаций на ранних этапах [4].

Тем не менее, с развитием автоматизации неизбежно возникают вызовы и риски. Технические сбои и недостатки в системах автоматизации могут привести к серьезным последствиям, вплоть до аварийных ситуаций. Кроме того, кибербезопасность становится все более актуальной проблемой в контексте автоматизированных систем, требуя постоянного мониторинга и защиты от кибератак [5]. Необходимо также учитывать, что автоматизация может вызвать сокращение рабочих мест и изменение роли человека в производственных процессах, что, в свою очередь, может повлиять на аспекты безопасности [6].

Параллельно с безусловными преимуществами автоматизации в обеспечении безопасности нефтегазовой промышленности, этот процесс также сопряжен с рядом вызовов и потенциальных рисков. Технические сбои и недостатки систем автоматизации могут иметь серьезные последствия для безопасности. Например, неправильная калибровка сенсоров или сбой в алгоритмах управления могут привести к неадекватным реакциям на аварийные ситуации [7]. Также, несмотря на все усилия в обеспечении кибербезопасности, автоматизированные системы остаются уязвимыми для кибератак, что может привести к неконтролируемым операциям и потенциальным авариям [8].

Одним из основных вызовов является баланс между автоматизацией и человеческим фактором. Процесс автоматизации может внести неопределенность в структуру рабочих мест, что может в свою очередь повлиять на общий уровень безопасности. Потеря определенных рабочих мест или неудачное внедрение автоматизации может увеличить риск, поскольку человеческая интуиция и мастерство остаются несравненными в ряде ситуаций [9].

С другой стороны, обучение и подготовка персонала становятся критически важными в условиях автоматизации. Операторы и инженеры должны обладать навыками, необходимыми для мониторинга и управления автоматизированными системами, а также для эффективного вмешательства в случае возникновения непредвиденных ситуаций [10]. Человеческое мастерство и интуиция остаются непревзойденными в способности реагировать на сложные и динамичные ситуации, что делает обучение персонала важным компонентом обеспечения безопасности.

Практика успешной интеграции человека и технологий в безопасность нефтегазовой промышленности демонстрирует, что сбалансированный подход имеет важное значение. Например, операторы и автоматизированные системы могут работать в партнерстве, используя преимущества обоих. В таком сотрудничестве автоматизированные системы выполняют монотонные или опасные задачи, а человек принимает решения в сложных и нестандартных ситуациях [11].

Технологические инновации намечают значительные изменения в сфере безопасности и автоматизации. Развитие сенсорных систем, искусственного интеллекта и аналитики данных позволит создавать более точные и быстрые системы мониторинга и реагирования на аварийные ситуации. Применение роботизации и автономных устройств позволит выполнять опасные задачи без риска для человеческой жизни .

Тенденции в развитии систем безопасности оказывают сильное влияние на нефтегазовую промышленность. Эволюция системы мониторинга и контроля позволяет операторам реагировать на изменяющиеся условия операций, улучшая безопасность и предотвращая аварии. Переход к цифровым платформам и облачным решениям обеспечивает более эффективное управление данными и оперативную обратную связь, что способствует более надежной безопасности.

Подведение итогов подчеркивает важность автоматизации в обеспечении безопасности нефтегазовой промышленности. Автоматизация позволяет минимизировать риски человеческого фактора и улучшать операционные стандарты, снижая вероятность человеческих ошибок. Однако важно поддерживать баланс между автоматизацией и ролью человека, так как уникальные навыки и опыт человека всегда будут ценными в обеспечении безопасности.

Список литературы

1. Смит Дж., Петров В. (2020). Роль автоматизации в повышении безопасности нефтяной и газовой промышленности. Журнал инженерной безопасности, 7 (1), С.15-21.
2. Браун А., Джонсон Л. (2018). Сенсорные системы для раннего обнаружения опасных условий на объектах нефтяной и газовой промышленности. Международный журнал промышленной безопасности, 34 (2), С.89-97.

3. Уильямс С., Миллер Р. (2019). Автоматизированные системы пожаротушения на нефтеперерабатывающих заводах. Технология безопасности сегодня, 6 (3), С.45-53.
4. Чен Л., Ли С. (2021). Прогнозная аналитика для повышения безопасности при автоматизированной добыче нефти. Журнал нефтяных технологий, 73 (5), С.68-75.
5. Джонс М., Смит Д. (2017). Проблемы кибербезопасности и решения для автоматизированных нефтегазовых операций. Кибербезопасность в энергетическом секторе: ежегодный отчет, 2017, С.25-36.
6. Гарсия Р., Эрнандес М. (2019). Влияние автоматизации на безопасность труда в нефтяной и газовой промышленности. Человеческий фактор в промышленности и обрабатывающей промышленности, 42 (4), С.512-520.
7. Johnson R., Miller A. (2020). Technical Failures in Automation and Their Impact on Safety. Journal of Safety Engineering, 8(2), pp.45-53.
8. Martinez E., Smith L. (2018). Cybersecurity Challenges in the Context of Automated Systems in the Oil and Gas Industry. International Journal of Industrial Security, 36(4), pp.67-75.
9. Brown J., Williams M. (2019). Impact of Automation on Employment and Its Implications for Safety. Human Factors in Industry and Manufacturing, 43(1), pp.102-110.
10. Clark A., White B. (2021). Training Strategies for Personnel Operating Automated Systems in Oil and Gas. Journal of Petroleum Technology, 75(3), pp.88-95.
11. Garcia R., Johnson D. (2017). Human-Machine Collaboration for Enhanced Safety in Oil and Gas Operations. Safety Technology Today, 4(2), pp.32-40.

References

1. Smith J., Petrov V. (2020). Role of Automation in Enhancing Oil and Gas Industry Safety. Journal of Safety Engineering, 7(1), pp.15-21.
2. Brown A., Johnson L. (2018). Sensor Systems for Early Detection of Hazardous Conditions in Oil and Gas Facilities. International Journal of Industrial Safety, 34(2), pp.89-97.
3. Williams C., Miller R. (2019). Automated Fire Suppression Systems in Oil Refineries. Safety Technology Today, 6(3), pp.45-53.
4. Chen L., Lee S. (2021). Predictive Analytics for Safety Enhancement in Automated Oil Production. Journal of Petroleum Technology, 73(5), pp.68-75.
5. Jones M., Smith D. (2017). Cybersecurity Challenges and Solutions in Automated Oil and Gas Operations.CybersecurityintheEnergySectorAnnualReport, 2017, pp.25-36.
6. Garcia R., Hernandez M. (2019). Impact of Automation on Occupational Safety in the Oil and Gas Industry. Human Factors in Industry and Manufacturing, 42(4), pp.512-520.
7. Johnson R., Miller A. (2020). Technical Failures in Automation and Their Impact on Safety. Journal of Safety Engineering, 8(2), pp.45-53.
8. Martinez E., Smith L. (2018). Cybersecurity Challenges in the Context of Automated Systems in the Oil and Gas Industry. International Journal of Industrial Security, 36(4), pp.67-75.
9. Brown J., Williams M. (2019). Impact of Automation on Employment and Its Implications for Safety. Human Factors in Industry and Manufacturing, 43(1), pp.102-110.
10. Clark A., White B. (2021). Training Strategies for Personnel Operating Automated Systems in Oil and Gas. Journal of Petroleum Technology, 75(3), pp.88-95.

11. Garcia R., Johnson D. (2017). Human-Machine Collaboration for Enhanced Safety in Oil and Gas Operations. Safety Technology Today, 4(2), pp.32-40.

Котлов В. Н., Фирсова С. А. Исследование микросервисного подхода к разработке архитектуры программных систем // Международный журнал информационных технологий и энергоэффективности. – 2023. –
Т. 8 № 8(34) с. 37–47

Международный журнал информационных технологий и энергоэффективности

Сайт журнала:
http://www.openaccessscience.ru/index.php/ijcse/

УДК 004.9

ИССЛЕДОВАНИЕ МИКРОСЕРВИСНОГО ПОДХОДА К РАЗРАБОТКЕ АРХИТЕКТУРЫ ПРОГРАММНЫХ СИСТЕМ

Котлов В. Н., [1] Фирсова С. А.
*ФГБОУ ВО "НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ МОРДОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМ. Н.П. ОГАРЁВА", Саранск Россия (430005, Республика Мордовия, город Саранск, Большевистская ул., д.68)***,** *e-mail: [1]karpushkinasa@yandex.ru*

В статье описывается микросервисный подход к разработке программных систем. Указываются преимущества и ограничения данного подхода в сравнении с монолитной архитектурой. Приводится пример решения, основанного на применении микросервисной архитектуры, в котором демонстрируется настройка взаимодействия между двумя независимыми сервисами на основе протокола gRPC.

Ключевые слова: Разработка программных систем, монолитная архитектура, микросервисная архитектура, протокол удаленного вызова процедур, язык программирования C#.

RESEARCH OF MICROSERVICE APPROACH TO THE DEVELOPMENT OF SOFTWARE SYSTEMS ARCHITECTURE

Kotlov V. N., [1]Firsova S. A.,
"NATIONAL RESEARCH MORDOVIA STATE UNIVERSITY. N.P. OGAREVA", Saransk Russia (430005, Republic of Mordovia, Saransk city, Bolshevistskaya street, 68), e-mail: [1]karpushkinasa@yandex.ru

The article describes a microservice approach to the development of software systems. The advantages and limitations of this approach in comparison with monolithic architecture are indicated. An example of a solution based on the use of a microservice architecture is given, which demonstrates the configuration of interaction between two independent services based on the gRPC protocol.

Keywords: Software systems development, monolithic architecture, microservice architecture, remote procedure call protocol, C# programming language.

В современном информационном обществе, где технологические требования постоянно эволюционируют, разработка программных систем стала неотъемлемой составляющей бизнеса и технического прогресса. Однако с ростом сложности приложений и необходимостью удовлетворять различным потребностям пользователей, традиционные архитектуры программных систем сталкиваются с вызовами, которые оказывают ограничивающее влияние на их эффективность.

Монолитная архитектура является стандартной моделью разработки программного обеспечения, в которой разрабатываемая программная система представляет собой единую, крупную и автономную систему, функционирующую независимо от других приложений [1].

При данном подходе все бизнес-задачи объединяются в одной большой вычислительной сети с единой базой кода. Например, на рисунке 1 показано приложение, построенное на основе монолитной архитектуры, включающее пользовательский интерфейс (User Interface), уровень доступа к данным (Data Access Layer), бизнес-логику приложения (Business Logic), которые функционируют в одной вычислительной сети. Данное приложение взаимодействует с базой данных, расположенной, возможно, в другой вычислительной сети.

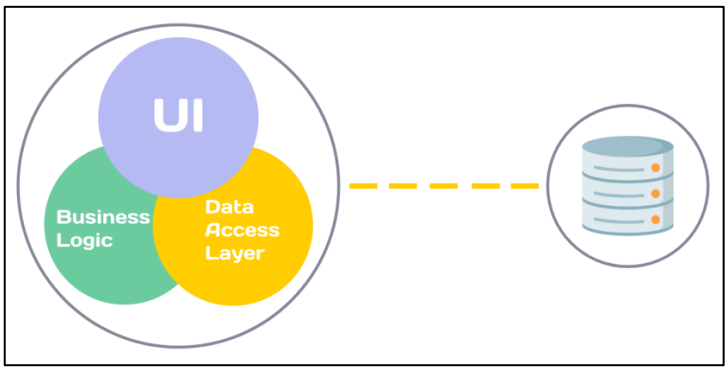


Рисунок 1 – Монолитная архитектура программной системы
Источник: Авторский материал

Монолитную архитектуру удобно применять на ранних этапах работы над программными проектами, чтобы облегчить развертывание программной системы и избежать излишних трудозатрат при управлении кодом [2]. Она позволяет сразу представить весь функционал приложения в одной монолитной сущности.

Преимущества монолитной архитектуры приведены в Таблице 1:

Таблица 1 – Преимущества монолитной архитектуры

Название	**Описание**
Простое развертывание	Использование единого исполняемого файла или каталога упрощает процесс развертывания приложения
Упрощенная разработка	Приложение проще разрабатывать, когда оно создано с использованием единой базы кода
Высокая производительность	Централизованная база кода и репозиторий позволяют одному интерфейсу API выполнять функции, которые в микросервисах могут выполнять многочисленные API
Упрощенное тестирование	Монолитное приложение представляет собой единую централизованную систему, что упрощает проведение сквозного тестирования по сравнению с распределенными приложениями
Удобная отладка	Весь код находится в одном месте, что упрощает поиск проблем и проведение запросов

Источник: Авторский материал

Тем не менее, монолитная архитектура также имеет свои недостатки и ограничения,

которые представлены в Таблице 2:

Таблица 2 – Недостатки монолитной архитектуры

Название	**Описание**
Снижение скорости разработки	Большое монолитное приложение усложняет и замедляет процесс разработки, особенно при работе в команде
Тяжелая масштабируемость	Невозможно масштабировать отдельные компоненты монолитной архитектуры, что может стать проблемой при обработке больших объемов данных и нагрузок на систему
Слабая отказоустойчивость	Ошибка в одном модуле может повлиять на доступность всего приложения, что делает монолит менее отказоустойчивым
Сложность внедрения новых технологий	Любые изменения в инфраструктуре или языке разработки сказываются на приложении целиком, что может привести к дополнительным затратам при внесении изменений в код
Ограниченная гибкость	Возможности монолитных приложений ограничены используемыми технологиями, что может ограничить применение инновационных технологий
Необходимость полного развертывания	Внесение даже небольших изменений потребует повторного развертывания всего монолитного приложения, что может замедлить процесс его обновления

Источник: Авторский материал

В поисках более эффективных решений и преодоления ограничений монолитной архитектуры, разработчики все чаще обращают взор к микросервисной архитектуре. Микросервисная архитектура представляет собой современный метод организации архитектуры программного обеспечения, который строится на основе независимо развертываемых служб, называемых микросервисами [3]. Каждый микросервис выполняет конкретную функцию и может работать автономно, имея свою базу данных и собственную бизнес-логику (Рисунок 2). Это означает, что каждая служба в микросервисной архитектуре может разрабатываться, обновляться, тестироваться и масштабироваться независимо от других служб. Взаимодействие между микросервисами происходит посредством четко определенных интерфейсов, часто использующих легковесные протоколы, такие как REST API или сообщений на основе очередей.

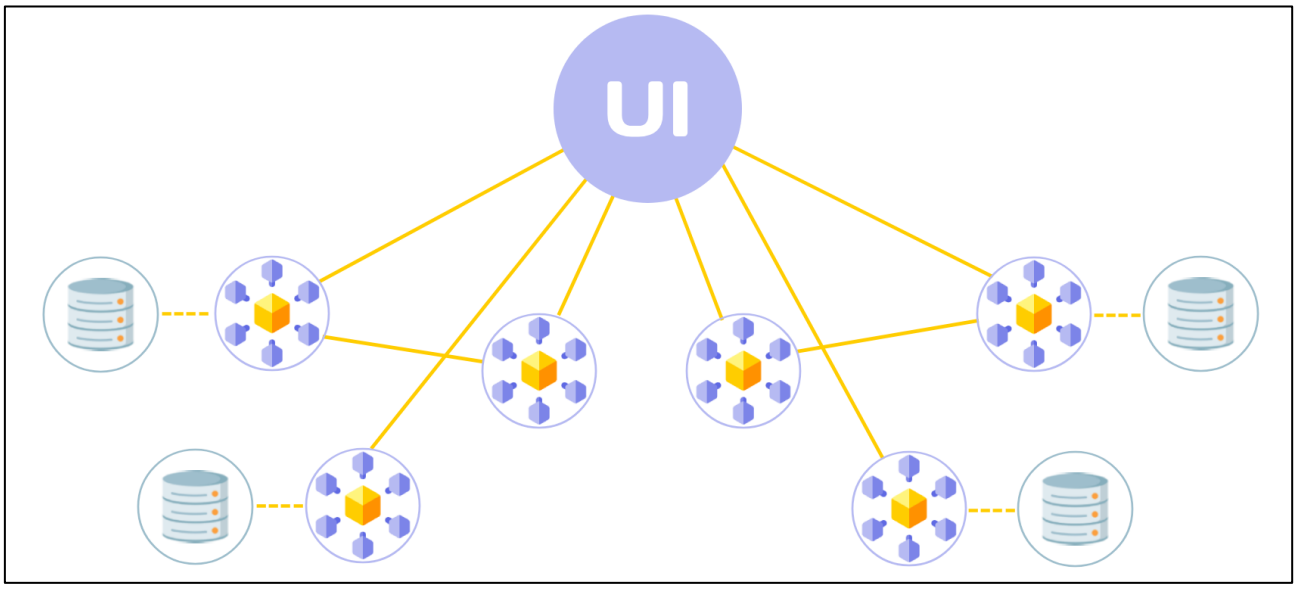


Рисунок 2 – Микросервисная архитектура программной системы

Источник: Авторский материал

Преимущества микросервисной архитектуры включают (Таблица 3):

Таблица 3 – Преимущества микросервисной архитектуры

Название	**Описание**
Гибкость разработки	Каждая служба может быть разработана независимо от других служб, что упрощает и ускоряет процесс разработки
Легкость масштабирования	При увеличении нагрузки на конкретную службу, ее можно масштабировать отдельно от других служб, что позволяет оптимизировать всевозможные ресурсы, затрачиваемые на масштабирование
Улучшенная отказоустойчивость	При отказе одной из служб, другие службы продолжат функционировать нормально, что повышает надежность всей системы
Легкость внедрения новых технологий	Каждая служба может использовать собственные технологии, что упрощает внедрение новых технологий в систему
Улучшенная организация команды	Команды разработчиков могут быть специализированы на определенных службах, что способствует повышению производительности и способности быстро реагировать на изменения

Источник: Авторский материал

При использовании микросервисной архитектуры следует учитывать следующие особенности её применения (Таблица 4):

Таблица 4 – Особенности применения микросервисной архитектуры

Название	Описание
Управление сложностью	Разделение сложных систем на множество микросервисов (служб) требует более тщательного управления и координации между ними
Сетевое взаимодействие	Взаимодействие между службами может вызывать задержки и проблемы с сетью передачи данных
Мониторинг и отладка	Необходимо обеспечить эффективный мониторинг и отладку каждой службы для обнаружения возможных проблем
Согласованность данных	При наличии нескольких баз данных в разных службах, обеспечение согласованности данных становится более сложной задачей
Управление версиями	Необходимо тщательно управлять версиями каждой из служб

Источник: Авторский материал

В целом, микросервисная архитектура является мощным инструментом для построения гибких и масштабируемых программных систем. Однако ее успешная реализация требует тщательного планирования, анализа и учета всех ограничений и особенностей разрабатываемого программного продукта. Выбор между монолитной и микросервисной архитектурой должен основываться на конкретных требованиях проекта и его потенциальных перспективах. Важно принимать во внимание, что микросервисы лучше подходят для крупных и сложных систем, где разделение функционала на независимые компоненты обеспечивает более эффективную разработку и масштабируемость. Небольшие проекты или проекты, в которых достижение высокой производительности не является первостепенной задачей, могут извлекать больше пользы из монолитной архитектуры.

Рассмотрим методы коммуникации между службами микросервисной архитектуры.

RESTful API (Representational State Transfer) **–** один из наиболее популярных способов взаимодействия между микросервисами. RESTful API основан на стандартных HTTP методах, таких как **GET** (получение данных), **POST** (создание новых данных), **PUT** (обновление данных) и **DELETE** (удаление данных). Каждый микросервис предоставляет свои конечные точки (endpoints), через которые другие службы могут обращаться для получения или обновления информации с помощью Json-объектов (рисунок 3):

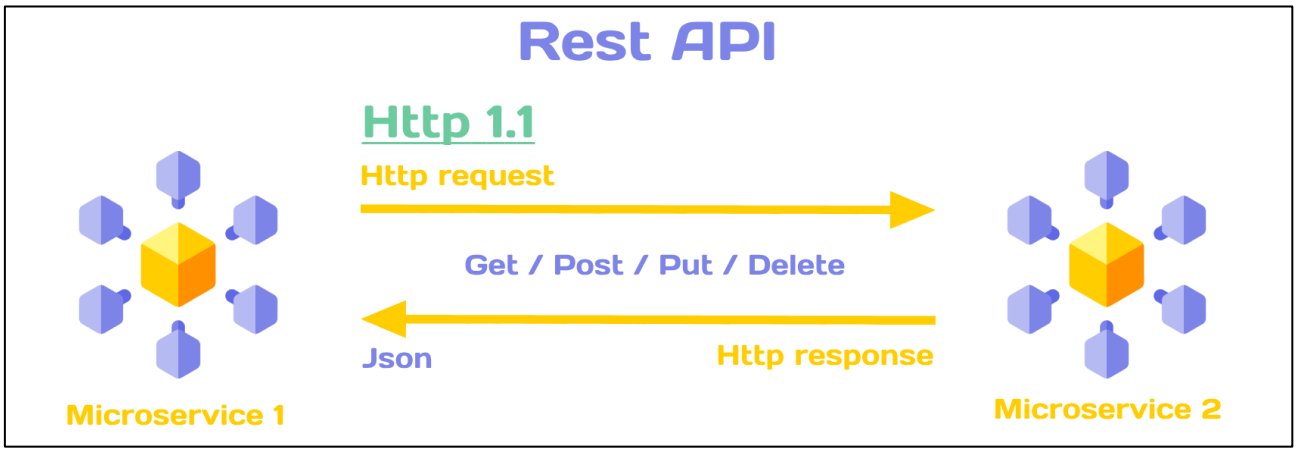


Рисунок 3 – Общение между микросервисами с помощью Rest API
Источник: Авторский материал

gRPC – современный протокол удаленного вызова процедур, разработанный компанией Google. Он основан на Protocol Buffers (protobufs) и предоставляет высокопроизводительный и эффективный способ взаимодействия между микросервисами. gRPC обеспечивает поддержку множества языков программирования и позволяет определить много различных типов данных. В отличие от REST API, протокол gRPC не предоставляет базовых методов, таких как GET, POST, PUT и DELETE. Вместо этого он использует механизмы RPC (Remote Procedure Call – удаленный вызов процедур), который позволяет клиентам вызывать методы, расположенные на удаленных серверах, так будто они вызывают локальные функции. Данные методы описываются в proto-файле и могут иметь различные типы (Рисунок 4): Unary (одинарный), Server Stream (серверный стриминг), Client Stream (клиентский стриминг) и Bi-directional stream (двунаправленный стриминг).

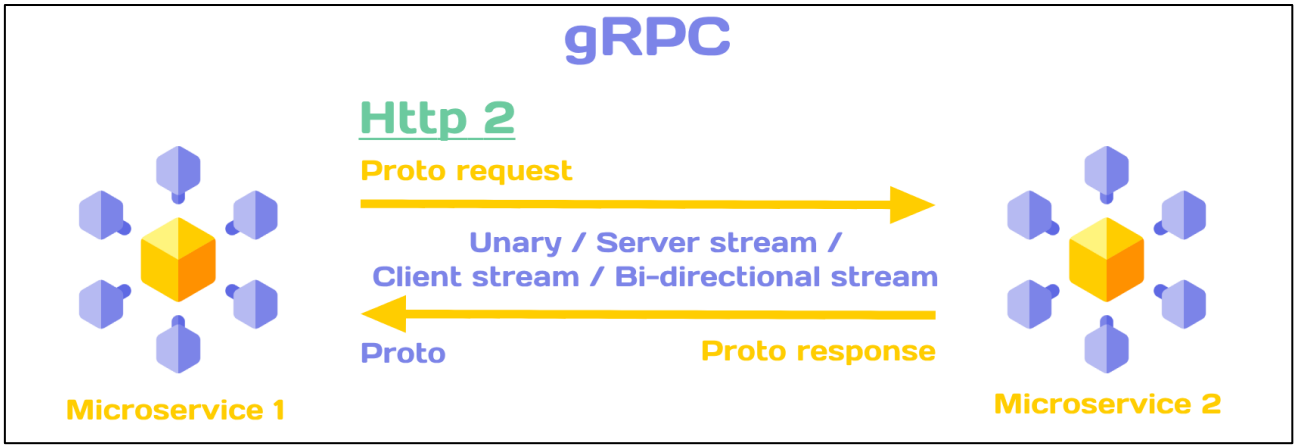


Рисунок 4 – Общение между микросервисами с помощью gRPC
Источник: Авторский материал

GraphQL – язык запросов для API, позволяющий клиентам запрашивать только те данные, которые им нужны, а не все данные, предоставляемые сервером. GraphQL позволяет определить точные требования клиентов к данным, что способствует оптимизации и

эффективности выполнения запросов. GraphQl передаёт данные с помощью объектов Json для выполнения запросов используются два типа методов: Query – запрос без изменения данных и Mutation – запрос с изменением данных (Рисунок 5).

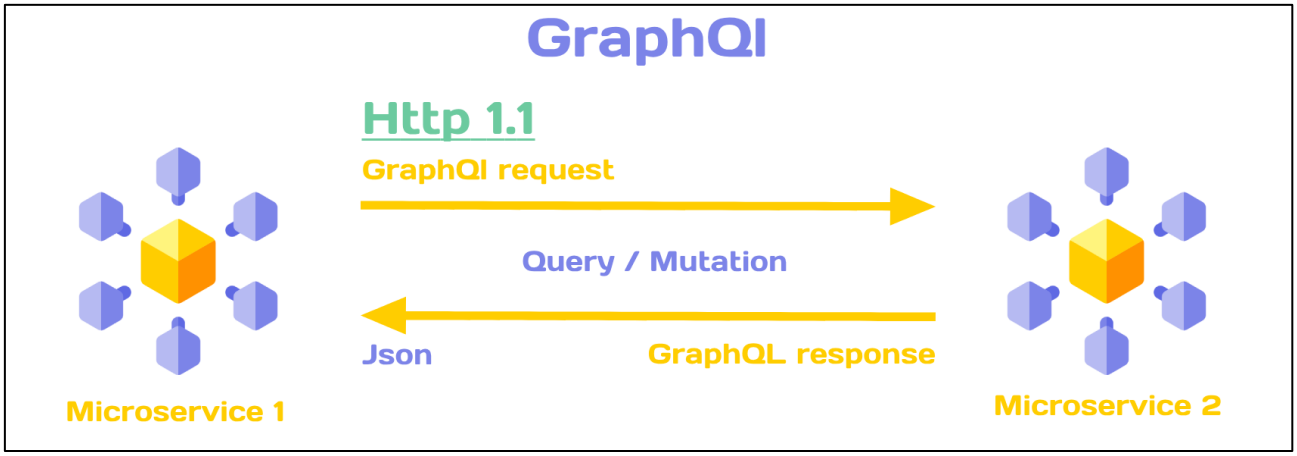


Рисунок 5 – Общение между микросервисами с помощью GraphQL
Источник: Авторский материал

Все перечисленные выше методы представляют синхронный способ обмена сообщениями. Данный способ применяется тогда, когда нужно получить ответ от сервера максимально быстро. Если скорость ответа или даже сам ответ от сервера нас не интересует, можно прибегнуть к асинхронному варианту взаимодействия микросервисов.

Асинхронный обмен сообщениями — это способ обмена сообщениями между микросервисами, в котором микросервисы общаются через через брокеры сообщений, такие как RabbitMQ или Apache Kafka. Это позволяет реализовать более слабую связь между компонентами системы и улучшить масштабируемость. Сервис, производящий сообщения, называется Producer, с помощью метода Produce передаёт сообщения в брокер сообщений. Сервис, принимающий сообщения, называется Consumer, с помощью команды Consume эти сообщения получает. Между сервисами и брокером сообщения передаются с помощью Http 1.1 в виде Json-объектов (Рисунок 6).

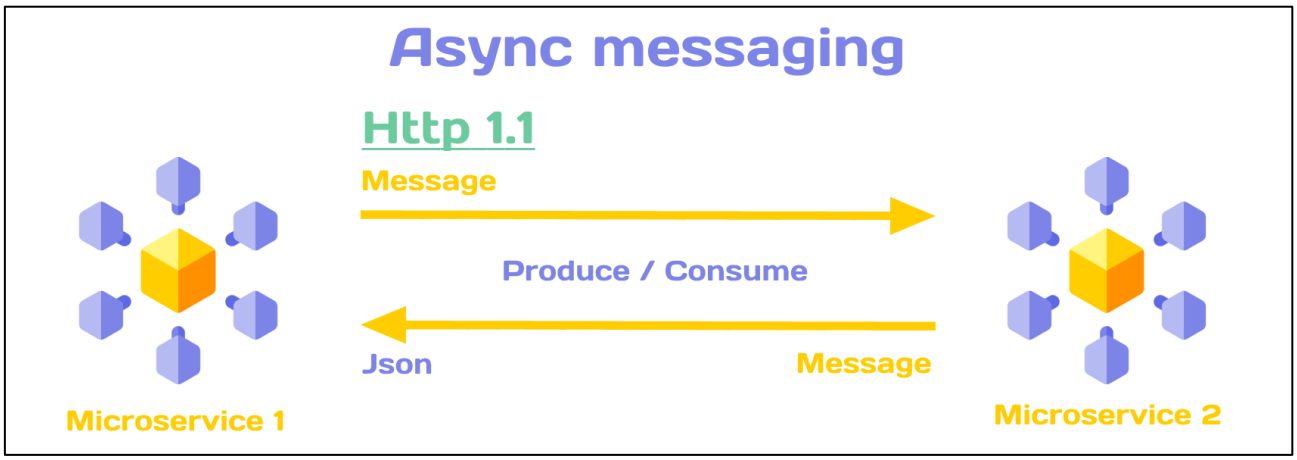


Рисунок 6 – Общение между микросервисами с помощью брокеров сообщений

Источник: Авторский материал

Важно отметить, что каждый способ коммуникации имеет свои преимущества и подходит для различных сценариев использования. Выбор способа коммуникации зависит от требований к разрабатываемой системе, типа передаваемых и получаемых данных, а также предпочтений и опыта команды разработчиков.

Приведем пример реализации микросервисной архитектуры на языке программирования C#. Для этого реализуем два микросервиса: первый будет отвечать за хранение информации о товарах, а второй – запрашивать эту информацию для последующей обработки и вывода полученных данных в консоль. Микросервисы будут взаимодействовать между собой через протокол gRPC.

Создадим новое решение с именем **MicroservicesExample.sln**, содержащее два отдельных проекта: **GoodsService,cs**, отвечающий за хранение данных о товарах, и **DisplayService.cs**, который будет запрашивать необходимую информацию о товарах у **GoodsService**, обрабатывать полученные данные и выводить их в консоль.

Перейдем к созданию proto-файла в проекте **GoodsService**, который будет описывать API для взаимодействия с этим сервисом. proto-файл представляет собой документ, содержащий описание всех доступных методов обмена сообщениями в рамках gRPC для данного сервиса. Это позволяет установить более эффективное взаимодействие между клиентскими и серверными приложениями, упрощая создание и обработку сообщений.

Кроме того в proto-файле определим методы обмена сообщениями с помощью протокола Buffers (protobuf), который является языком для определения структуры данных и сервисов взаимодействия с ними.

На Рисунке 7 представлена структура proto-файла для микросервиса **GoodsService**, который содержит один метод **GetGood**. Данный метод принимает запрос типа **GetGoodRequest**, содержащий идентификатор товара, и возвращает ответ типа **GetGoodResponse**, содержащий объект "Good", в котором содержится информация о товаре:

```
syntax = "proto3";
package GrpcGoodsService;
service GrpcGoodsService
{ rpc GetGood (GetGoodRequest) returns (GetGoodResponse) {} }

message Good
{ string Title = 1; }

message GetGoodRequest
{ int64 id = 1; }

message GetGoodResponse
{ Good good = 1; }
```

Рисунок 7 – Proto-файл микросервиса GoodsService

Источник: Авторский материал

После компиляции данного proto-файла будет получен полноценный C#-сервис, который

можно унаследовать и переопределить его методы для добавления необходимой логики обработки запросов и ответов.

На Рисунке 8 показано наследование автосгенерированного из proto-файла сервиса **GoodsService**. Как видно из рисунка, у нас появился описанный в proto-файле метод **GetGoods**, который получает на вход **GetGoodsRequest**, а отдает **GetGoodResponse**, содержащий объект Good. Для примера был добавлен объект Goods, который будет содержать информацию о товарах. Метод **GetGood** будет возвращать один из его объектов или ошибку, если индекс объекта находится за пределами массива.

```
     public class GoodsService :
GrpcGoodsService.GrpcGoodsService.GrpcGoodsServiceBase
{
    private static readonly string[] Goods = { "Кружка", "Рюкзак", "Ноутбук",
"Тетрадь", "Проектор" };

         public override async Task<GetGoodResponse> GetGood(GetGoodRequest
request, ServerCallContext context)
    {
        if (request.Id < 0 || request.Id > Goods.Length - 1)
           {throw new ArgumentOutOfRangeException(nameof(request.Id));}
            return new GetGoodResponse
            {
             Good = new Good { Title = Goods[request.Id] }
        };
    }
}
```

Рисунок 8 – Наследование автосгенерированного класса
Источник: Авторский материал

Теперь можно приступить к написанию клиента. Для этого мы должны скопировать в **DisplayService** вышеприведенный proto-файл. После выполнения сборки проекта **DisplayService**, аналогично проекту **GoodsService**, появится класс **GrpcGoodsServiceClient**, с помощью которого можно вызывать методы из сервиса **GoodsService** (Рисунок 9).

```
     static async Task
HandleGrpc(GrpcGoodsService.GrpcGoodsService.GrpcGoodsServiceClient client)
{
         Console.WriteLine($"Запрос к серверу: Id = 1");
         var response = await client.GetGoodAsync(new GetGoodRequest { Id = 1
});
         Console.WriteLine($"Ответ сервера: {response.Good.Title}");
}
```

Рисунок 9 – Метод GetGoodsAsync класса GrpcGoodsServiceClient
Источник: Авторский материал

Вызвав метод GetGoodsAsync внутри функции Main (Рисунок 10), мы получим результат, показанный на рисунке 11. Также на этом рисунке представлена архитектура

созданного решения.

```
     public static async Task Main()
{
    var host = await Startup.CreateHost();
    var grpcClient = host.Services.GetRequiredService
            <GrpcGoodsService.GrpcGoodsService.GrpcGoodsServiceClient>();
         Console.WriteLine($"Запрос к серверу: Id = 1");
    var response = await grpcClient.GetGoodAsync(new GetGoodRequest { Id = 1
});
    Console.WriteLine($"Ответ сервера: {response.Good.Title}");
}
```

Рисунок 10 – Вызов метода GetGoodsAsync класса GrpcGoodsServiceClient
Источник: Авторский материал

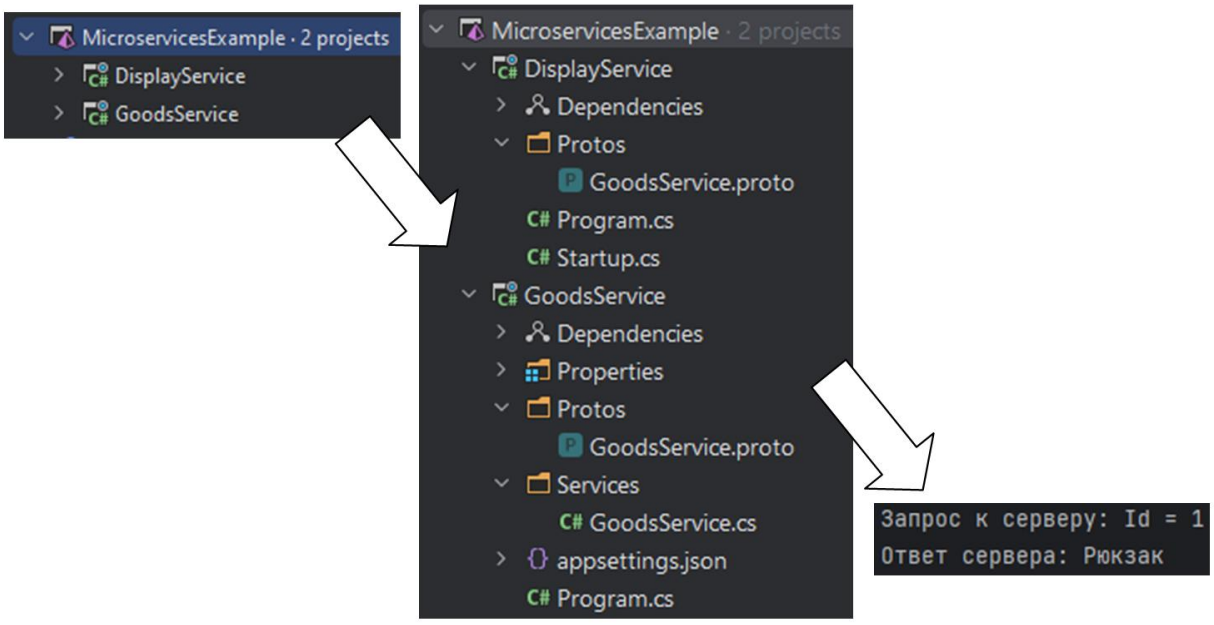


Рисунок 11 – Архитектура созданного решения и результат выполнения
Источник: Авторский материал

Таким образом, на данном примере было показано, как настроить взаимодействие между двумя независимыми сервисами на основе протокола gRPC.

Заключение.

Микросервисная архитектура представляет собой мощный и инновационный подход к разработке программного обеспечения, который стремительно завоевывает популярность в сфере информационных технологий. Её преимущества включают гибкость, масштабируемость, улучшенный процесс разработки и отказоустойчивость. Однако, её внедрение требует дополнительных усилий, хорошего понимания современных технологий и умения обеспечивать безопасность системы.

Список литературы

1. Никитин, И. В. Сравнение подходов монолитной архитектуры и микросервисной архитектуры при реализации серверной части веб-приложения / И. В. Никитин, Т. Ю. Гриценко // Дневник науки. – 2020. – № 3(39). – 22с.
2. Надейкина, Л. А. Распределенные системы, построенные на базе микросервисной архитектуры / Л. А. Надейкина, Н. И. Черкасова // Инновационные, информационные и коммуникационные технологии. – 2019. – № 1. – С. 300-304.
3. Караханова, А. А. Анализ микросервисной архитектуры, монолитных приложений, архитектуры SOA / А. А. Караханова // Синергия Наук. – 2020. – № 46. – С. 255-262.

References

1. Nikitin, I. V. Comparison of approaches of monolithic architecture and microservice architecture in the implementation of the server part of a web application / I. V. Nikitin, T. Y. Gritsenko // Diary of Science. – 2020. – № 3(39). – p. 22.
2. Nadeikina, L. A. Distributed systems based on microservice architecture / L. A. Nadeikina, N. I. Cherkasova // Innovative, information and communication technologies. – 2019. – No. 1. – pp. 300-304.
3. Karakhanova, A. A. Analysis of microservice architecture, monolithic applications, SOA architecture / A. A. Karakhanova // Synergy of Sciences. – 2020. – No. 46. – pp. 255-262.

Сафин М.А., Сафиуллина А.Ф. Экологические аспекты автоматизации в нефтегазовой промышленности // Международный журнал информационных технологий и энергоэффективности. – 2023. –
Т. 8 № 8(34) с. 48–51

Международный журнал информационных технологий и энергоэффективности

Сайт журнала:
http://www.openaccessscience.ru/index.php/ijcse/

УДК 004.9

ЭКОЛОГИЧЕСКИЕ АСПЕКТЫ АВТОМАТИЗАЦИИ В НЕФТЕГАЗОВОЙ ПРОМЫШЛЕННОСТИ

Сафин М.А., [1]Сафиуллина А.Ф.
ФГБОУ ВО "КАЗАНСКИЙ ГОСУДАРСТВЕННЫЙ ЭНЕРГЕТИЧЕСКИЙ УНИВЕРСИТЕТ", Казань Россия (420066, Республика Татарстан, город Казань, Красносельская ул, д. 51), *e-mail: [1]alsukizikay@gmail.com*

Данная статья рассматривает вопросы экологической устойчивости нефтегазовой промышленности. Проводится анализ того, как автоматизированные системы могут сыграть роль в борьбе с негативными последствиями на окружающую среду. В работе приводятся примеры различных методов автоматизации нефтегазовой отрасли такие, как мониторинг, сенсорные системы, аналитика данных и т.д.

Ключевые слова: Экология, автоматизация, нефтегазовая отрасль, устойчивое развитие.

ENVIRONMENTAL ASPECTS OF AUTOMATION IN THE OIL AND GAS INDUSTRY

Safin M.A., **Safiullina A.F**.
KAZAN STATE POWER ENGINEERING UNIVERSITY, Kazan, Russia (420066, Republik of Tatarstan, Kazan city, Krasnoselskaya street, 51), e-mail: [1]alsukizikay@gmail.com

This article examines the issues of environmental sustainability of the oil and gas industry. The analysis of how automated systems can play a role in the fight against negative consequences on the environment is carried out. The paper provides examples of various methods of automation of the oil and gas industry, such as monitoring, sensor systems, data analytics, etc.

Keywords: Ecology, automation, oil and gas industry, sustainable development.

В современном мире экологические аспекты занимают центральное место в дискуссиях о будущем промышленных отраслей. В этом контексте нефтегазовая промышленность играет особую роль, так как она является ключевым источником энергии для мировой экономики. Эта отрасль обеспечивает не только энергетическую безопасность, но и важную составляющую национальных бюджетов многих стран. Однако, сопутствующие процессы добычи, переработки и транспортировки нефти и газа оказывают негативное воздействие на окружающую среду.

Экологические вопросы привлекают все большее внимание современной общественности и бизнеса. Поддержание экологической устойчивости становится приоритетом, особенно в свете изменения климата и экологических катастроф. Экологически ответственный бизнес и индустрия в целом становятся важными компонентами общественной ответственности и устойчивого развития.

Нефтегазовая промышленность, несмотря на свой важный вклад в мировую экономику, также сопровождается серьезными экологическими последствиями. Выбросы парниковых газов, загрязнение водных ресурсов, разрушение экосистем — все это вызывает обеспокоенность и требует принятия мер для снижения отрицательного воздействия.

В данной статье мы рассмотрим экологические аспекты в контексте нефтегазовой промышленности и исследуем, как автоматизация может стать ключевым инструментом в борьбе с негативными экологическими последствиями. Мы проанализируем важность соблюдения баланса между экономическими интересами и заботой о природной среде, а также рассмотрим технологические решения и будущие перспективы, направленные на обеспечение экологической устойчивости в нефтегазовой промышленности.

Автоматизация играет ключевую роль в снижении негативного экологического воздействия, связанного с нефтегазовой промышленностью. Современные технологии автоматизации находят широкое применение в различных процессах этой отрасли, позволяя эффективно управлять производственными операциями и сокращать негативные экологические последствия.

Автоматизированные системы и технологии внедряются на всех этапах нефтегазового производства — от добычи и транспортировки до переработки. С помощью автоматического мониторинга и управления можно точно контролировать параметры производственных процессов, что приводит к снижению выбросов вредных веществ и энергопотребления [1]. Применение автоматизации в оптимизации расходов воды, энергии и других ресурсов позволяет добиваться более эффективного их использования, снижая негативное воздействие на окружающую среду [2].

Применение автоматизации также способствует оптимизации производственных процессов, что в свою очередь снижает потери и улучшает эффективность добычи и переработки. Точное управление параметрами позволяет минимизировать потери и избыточное использование ресурсов, что положительно сказывается на экологической устойчивости [3].

Важным аспектом в контексте безопасности окружающей среды является предотвращение аварий и инцидентов. Автоматизация позволяет создавать системы мониторинга и предупреждения, которые могут своевременно реагировать на потенциально опасные ситуации и предотвращать экологические катастрофы. Примером такой системы может служить автоматическое обнаружение утечек нефти или газа и немедленное принятие мер для их ликвидации [4]. Автоматизация является эффективным инструментом для снижения негативного экологического воздействия нефтегазовой промышленности. Оптимизация производственных процессов, контроль параметров и предотвращение аварийных ситуаций способствуют улучшению экологической устойчивости этой важной отрасли

Использование сенсорных сетей и аналитики данных играет значительную роль в улучшении экологических аспектов нефтегазовой промышленности. Сенсорные системы представляют собой сети датчиков, предназначенных для непрерывного мониторинга окружающей среды и параметров производственных процессов. Они позволяют оперативно

отслеживать уровни загрязнений, температуру, давление и другие показатели, что способствует более точному контролю над экологическими параметрами [5].

Дополнительно, аналитика данных играет важную роль в обработке информации, полученной от сенсорных систем. С помощью специализированных алгоритмов и программных решений можно выявлять аномалии и изменения в окружающей среде, что позволяет быстро реагировать на потенциальные угрозы и предотвращать экологические аварии [6]. Аналитика данных также способствует оптимизации производственных процессов, что влечет за собой снижение негативного воздействия на окружающую среду.

Экологические инновации в области автоматизации представляют собой важную перспективу для будущего нефтегазовой промышленности. Прогнозируется развитие технологий, направленных на еще более эффективное снижение выбросов вредных веществ, оптимизацию потребления ресурсов и предотвращение экологических происшествий [7]. Технологии, такие как автоматизированные системы улавливания выбросов и использования альтернативных источников энергии, могут значительно способствовать улучшению экологической устойчивости отрасли.

Важным аспектом в данной области является сбалансированный подход к экологической автоматизации. При внедрении новых технологий необходимо учитывать не только экологические аспекты, но и бизнес-эффективность. Государственные регулирования и корпоративная ответственность играют ключевую роль в поддержании этого баланса, обеспечивая соблюдение экологических стандартов и требований безопасности [8].

Роль автоматизации в снижении экологического воздействия является неоспоримой. Автоматизированные системы и технологии позволяют более точно контролировать параметры производства, своевременно реагировать на изменения в окружающей среде и минимизировать негативные последствия. Примеры успешного использования автоматизации для оптимизации расходов ресурсов и снижения выбросов вредных веществ подтверждают эффективность данного подхода.

Однако важно подчеркнуть, что успешная экологическая автоматизация требует сбалансированного подхода. Соблюдение баланса между экологическими аспектами и бизнес-эффективностью, а также соблюдение стандартов безопасности и государственных требований, остаются приоритетными задачами. Совместное усилие предприятий, государственных органов и научных сообществ в этой области позволит добиться наилучших результатов.

Таким образом, экологические аспекты и автоматизация тесно взаимосвязаны и влияют друг на друга. Современные технологии и инновации предоставляют возможности для более экологически устойчивой нефтегазовой промышленности. Сбалансированный подход к экологической автоматизации способствует созданию более безопасной, эффективной и ответственной отрасли, способной успешно справляться с вызовами современности и охранять окружающую среду для будущих поколений.

Список литературы

1. Смит Дж. и Джонсон М. (2020). Автоматизация и окружающая среда в нефтегазовом секторе. Промышленный робот: Международный журнал исследований и применения робототехники, 47 (2), С.170-176.

2. Ван, Ю., Тан, Х. К., и Ван, Х. (2021). Применение автоматизированных технологий в охране окружающей среды нефтяных месторождений. Исследование и эксплуатация энергии, 39 (2), С.789-804.
3. Гольшенас А. Н., Сюй С. и Дабабне А. С. (2020). Автоматизация нефтегазовой промышленности: современные области применения и тенденции. Журнал нефтяной науки и инженерии, 195, С.107984.
4. Алобайди Х., Хадианфард М. Дж. и Аль-Марзуки М. (2021). Интеллектуальная система обнаружения утечек с использованием беспроводных сенсорных сетей в нефтегазовой промышленности. Датчики, 21(9), С.3180.
5. Лю Х., и Чжао Дж. (2020). Мониторинг окружающей среды на основе датчиков в нефтяной и газовой промышленности. Журнал нефтяной науки и инженерии, 188, С.106834.
6. Srivastava, A., Garg, A., & Kumar, P. (2021). Big Data Analytics for Environmental Monitoring: A Review. Environmental Monitoring and Assessment, 193(2), C.72.
7. Lopes, F. M., Araujo, R. M., & Pinto, J. G. (2019). Smart Technologies for Environmentally Sustainable Oil and Gas Production.In Smart Technologies for Sustainable Smallholder Agriculture (pp. 205-226). Springer, Cham.
8. Azapagic, A., Perdan, S., & Clift, R. (2017). Sustainable development in the process industry: a general framework and a case study. Sustainable Development, 25(5), pp. 322-335.

References

1. Smith, J., & Johnson, M. (2020). Automation and the environment in the oil and gas sector. Industrial Robot: The International Journal of Robotics Research and Application, 47(2), pp. 170-176.
2. Wang, Y., Tan, H. Q., & Wang, X. (2021). Application of automation technology in oilfield environmental protection. Energy Exploration & Exploitation, 39(2), pp. 789-804.
3. Golshenas, A. N., Xu, C., &Dababneh, A. S. (2020). Automation of the oil and gas industry: Current applications and trends. Journal of Petroleum Science and Engineering, 195, pp.107984.
4. Alobaidy, H., Hadianfard, M. J., & Al-Marzouqi, M. (2021). Intelligent leak detection system using wireless sensor networks in the oil and gas industry. Sensors, 21(9), pp. 3180.
5. Liu, H., & Zhao, J. (2020). Sensor-based environmental monitoring in the oil and gas industry. Journal of Petroleum Science and Engineering, 188, pp.106834.
6. Srivastava, A., Garg, A., & Kumar, P. (2021). Big Data Analytics for Environmental Monitoring: A Review. Environmental Monitoring and Assessment, 193(2), pp.72.
7. Lopes, F. M., Araujo, R. M., & Pinto, J. G. (2019). Smart Technologies for Environmentally Sustainable Oil and Gas Production.In Smart Technologies for Sustainable Smallholder Agriculture (pp. 205-226). Springer, Cham.
8. Azapagic, A., Perdan, S., & Clift, R. (2017). Sustainable development in the process industry: a general framework and a case study. Sustainable Development, 25(5), pp. 322-335.

Шерстнев А.О. Гибкие методологии и современные инструменты для организации учебного процесса // Международный журнал информационных технологий и энергоэффективности. – 2023. – Т. 8 № 8(34) с. 52–59

Международный журнал информационных технологий и энергоэффективности

Сайт журнала:
http://www.openaccessscience.ru/index.php/ijcse/

УДК 004.9

ГИБКИЕ МЕТОДОЛОГИИ И СОВРЕМЕННЫЕ ИНСТРУМЕНТЫ ДЛЯ ОРГАНИЗАЦИИ УЧЕБНОГО ПРОЦЕССА

Шерстнев А.О.
Аккредитованное образовательное частное учреждение ВО "МОСКОВСКИЙ ФИНАНСОВО-ЮРИДИЧЕСКИЙ УНИВЕРСИТЕТ МФЮА", Москва, Россия (115191, город Москва, ул. Серпуховский Вал, д.17 к.1), e-mail: Sm_alex99@mail.ru

Работа посвящена анализу удаленной работы преподавателей и студентов при дистанционном обучении с выявлением достоинств и недостатков программных средств его обеспечения. Предложено использовать технологию Scrum для проведения дистанционного обучения, позволяющую разбивать образовательный процесс на отрезки определенной длины (спринты) и отслеживать выполнение заданий в соответствии с задаваемыми временными интервалами. Данный подход позволяет участникам учебного процесса адекватно контролировать выполнение учебного плана. Проведен сравнительный анализ платформ для онлайн-взаимодействия Zoom, Skype, Discord, BigBlueButton. Сделан вывод о предпочтительном использовании платформы Discord для семинарских и лабораторных занятий, а приложения Zoom – для лекционных. Обращено внимание на недостатки системы Moodle, которая используется во многих вузах. Полученные в работе результаты могут быть использованы при организации удаленного взаимодействия преподавателей и обучаемых.

Ключевые слова: Дистанционное обучение, преподаватель, студент, платформы дистанционного обучения, Scrum.

ANALYSIS OF REMOTE WORK OF A TEACHER USING FLEXIBLE APPROACHES TO THE ORGANIZATION OF THE EDUCATIONAL PROCESS

Sherstnev A.O.
Accredited private educational institution of Higher Education "Moscow University of Finance and Law", Moscow, Russia (115191, Moscow, Serpukhovsky Val str., 17 k.1), e-mail: Sm_alex99@mail.ru

The work is devoted to the analysis of remote work of teachers and students in remote training with the identification of advantages and disadvantages of software tools for its provision. It is proposed to use Scrum technology for distance learning, which allows you to divide the educational process into segments of a certain length (sprints) and track the completion of tasks in accordance with the specified time intervals. This approach allows the participants of the educational process to adequately monitor the implementation of the curriculum. A comparative analysis of the platforms for online interaction Zoom, Skype, Discord, BigBlueButton. The conclusion is made about the preferred use of the Dis-cord platform for seminars and laboratory classes, and the Zoom application for lectures. Attention is drawn to the shortcomings of the Moodle system, which is used in many universities. The results obtained in the work can be used in the organization of remote interaction between teachers and trainees.

Keywords: Distance learning, teacher, student, distance learning platforms, Scrum.

Введение

Актуальность вопросов организации дистанционного обучения, с учетом современной эпидемиологической обстановки, не вызывает сомнений. Высшие и средние учебные заведения стараются обеспечить преподавателей и обучаемых необходимым инновационным программным обеспечением как инструментом для проведения удаленных занятий. Тенденции внедрения информационных технологий в образовательный процесс наиболее развитых государств показывают, что заметные изменения в мировой системе образования являются реальностью. Кроме того, происходит техническое переоснащение учебных заведений. Одна из основных задач, стоящих перед преподавателями образовательных учреждений, заключается не только во внедрении дистанционного обучения в собственную методическую практику, но и в обеспечении его эффективности. Для этого необходимо использовать программное обеспечение, которое наиболее соответствует каждому виду учебных занятий и позволяет организовать удобное и эффективное взаимодействие между преподавателями и студентами. В настоящей работе дан сравнительный анализ наиболее распространенных программных платформ и инструментальных средств, используемых для дистанционного обучения в университетской среде.

1. Некоторые достоинства и недостатки дистанционного обучения

Один из возможных вариантов рекомендаций для работы преподавателя при дистанционном обучении предлагается ФГАОУ ВО Национальный исследовательский университет «Высшая школа экономики» [1]. Рекомендации сведены в Таблицу 1.

Таблица 1 – Рекомендации для работы преподавателя на дистанционном обучении

Рекомендации	Как делать не следует
Асинхронное обучение: преподаватель заранее разрабатывает задания для студентов и дает инструкции по использованию технических инструментов обучения.	Синхронное обучение: преподаватель читает лекции и ведет семинары только онлайн, не предоставляя студентам подготовительных материалов.
При онлайн-обучении освоение материала по различным причинам может занимать больше времени, поэтому необходимо заранее определить приоритетные темы и задания.	Давать задания на каждый день или устанавливать короткие сроки исполнения заданий.
Предоставление для студентов плана будущих занятий – темы, их последовательность, примерное время прохождения, используемые инструменты, четкие критерии системы оценивания.	Выдавать задания с неопределенными сроками исполнения. Чаще всего они оказываются невыполненными, а задания без жестких критериев – выполненными некачественно.
Создать форму для обратной связи для выяснения, какие темы и инструменты вызывают у студентов затруднения. Назначить регулярные часы приема, когда студенты смогут связаться с преподавателем для обсуждения материала.	Использовать много онлайн-чатов и отвечать на сообщения целый день. Если часы для консультаций будут выделены грамотно и в достаточном количестве, то студентам не будет необходимости писать в нерабочее время.

Представленные в Таблице 1 рекомендации можно расширить. В зарубежных школах и университетах применяется подход к обучению в формате Scrum [2-4]. При дистанционном обучении этот подход может оказаться достаточно эффективным с учетом того, что обучающимся намного сложнее поддерживать себя в мотивированном к учебе состоянии и соблюдать учебный график. Результаты опросов показывают [5], что у многих студентов день на дистанционном обучении выглядит так, как представлено в Таблице 2.

Таблица 2 – Распорядок дня студента при дистанционном обучении

Действия	Подъем	Попытка найти ссылку на занятие	Лекция в Zoom	Вопрос преподавателю	Перерыв	Семинар с добавленным материалом в Moodle
Отлично						
Хорошо						
Приемлемо						
	Выспался и успел позавтракать (т. к. не надо никуда ехать)	Не знает, где ссылка	Слушает лекцию	Получил моментальный ответ	Можно пообедать, не тратя времени в очереди в столовую	Получил задание по семинарскому занятию

Действия	Вопрос преподавателю на почту	Следующая лекция	Просмотр видео на Youtube вместо лекции	Обед	Семинар по программированию	Отправление выполненного задания преподавателю	Окончание занятий
Отлично							
Хорошо							
Прием-лемо							
	Долго не получает ответа, из-за этого не успевает сделать задание к сроку	Доделывает задание предыдущего семинара, пропускает половину лекции и решает больше ее не слушать	Доволен, потому что всё равно получит балл за посещение	Не надо ждать очередь в столовой	Получает задание, приступает к выполнению	Не получает никаких комментариев, не понимает, правильно ли он выполнил задание	Нет необходимости ехать в метро

Анализ приведенных в Таблице 2 данных показывает, что основной проблемой студента на дистанционном обучении является сложность поиска тематического материала для каждой

учебной дисциплины. Кроме того, нередко имеет место долгое ожидание ответов от преподавателя о времени проведения консультационных занятий и по вопросам домашних заданий. Данные проблемы можно решить с помощью существующих программных продуктов, однако каждый из них имеет свои недостатки. Для их выявления целесообразно выполнить сравнение планирования дня студента при очном посещении университета, один из вариантов которого приведен в Таблице 3, с распорядком дня при дистанционном обучении.

Таблица 3 – Распорядок дня студента при очном обучении

Действия	Подъем	Завтрак	Поездка в метро в университет	Нахождение на учебном занятии	Перерыв	Контрольная на следующем семинаре
Отлично						
Хорошо						
Приемлемо						
	Необходимо рано вставать, чтобы доехать до университета	Необходимо быстро выполнить все утренние дела, чтобы успеть в университет	Много людей в общественном транспорте, тратится много времени на дорогу	Полная концентрация на прослушивании лекции	Составленное расписание с короткими переменами не позволяет полноценно поесть	Возможно опоздание, если пойти в столовую
Действия	Сложное задание в контрольной работе	Сдача контрольной работы	Получение домашнего задания	Обед	Поездка на метро домой	Выполнение домашнего задания дома
Отлично						
Хорошо						
Приемлемо						
	Можно спросить у преподавателя	Понятно, как сдавать работу	Знает когда и куда сдавать	После занятий можно поесть, но придется задержаться	Много людей в общественном транспорте, тратится много времени на дорогу	Также можно выполнять задание дома при дистанционном обучении

В Таблицах 2 и 3 красные столбцы отображают, на каких этапах студент испытывает трудности. Они возникают, когда необходимо пообщаться с преподавателем, а также при поиске ссылки на занятие, потому что не во всех университетах принята достаточно

эффективная система управления обучением. В качестве примера системы управлением обучением можно привести Moodle [6]. Кроме того, дистанционное обучение снимает проблему траты времени на дорогу и позволяет вовремя обедать, потому что небольшой перерыв между занятиями в университете нередко не дает возможности посещения столовой без риска опоздания на занятия.

2. Анализ программного обеспечения для дистанционного обучения

Одной из самых популярных систем управления обучением, использующихся в университетах, является Moodle. Это LMS-платформа управления учебными курсами, которую можно установить на собственный сервер и интегрировать ее с другими сервисами университета. Moodle обеспечивает возможность создания курсов, размещения в них учебных материалов, выставления заданий с последующей загрузкой решений студентами и их оцениванием преподавателем. Однако у данной системы есть ряд недостатков, которые затрудняют ее использование. Например, нет возможности обмена сообщениями в чате: преподавателю в любом случае необходимо общаться со студентами в мессенджере или по электронной почте, но это приводит к увеличению количества сообщений на странице преподавателя. Moodle не решает проблему структурирования информации, хотя это и является одной из его задач. Интерфейс системы нельзя признать удобным и очевидным. Кроме того, шрифт экранных форм, в том числе профиля преподавателя, мелкий, как это видно на Рисунке 1. Отсутствует информация по взаимодействию с преподавателем, нет расписания его работы. Есть только список закрепленных курсов, но из него не следует, за какие именно из них преподаватель отвечает (как, например, лектор), а в каких просто принимает участие.

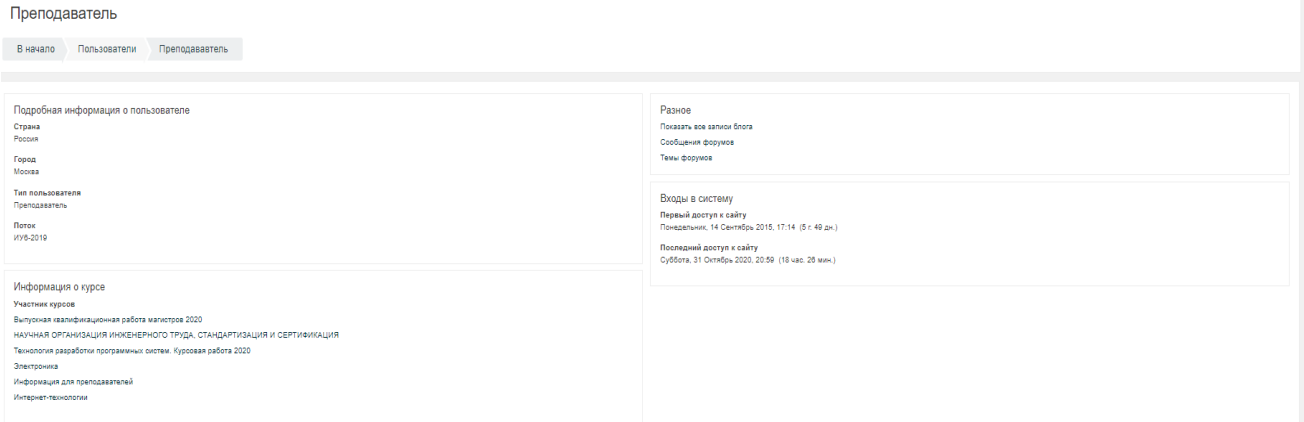


Рисунок 1 – Страница преподавателя в Moodle

Список домашних или контрольных заданий можно найти только на странице курса. То есть студенту не будут приходить оповещения о том, что появилось новое домашнее задание или контрольная работа. Эту информацию нужно предоставлять отдельно, поэтому возникает необходимость в мессенджере. Целесообразно иметь систему оповещения по всем видам домашних заданий и работ, а также единый чат, в который были бы подключены как преподаватели, так и обучаемые, однако в Moodle такой чат отсутствует.

В качестве альтернативы может быть использован Scrum-метод, который позволяет спланировать определенные отрезки времени и задания в их пределах. В этом случае студенту

не нужно помнить сроки сдачи заданий, поскольку данные отрезки заканчиваются в один и тот же день каждую неделю или две: это время устанавливается преподавателем. На специальной доске, состоящей из трех столбцов «Что нужно сделать», «В работе», «Сделано», отображается статус каждой задачи каждого студента. Такую доску можно вести онлайн и оффлайн. Общий список задач размещается в общем списке, и студенты всегда могут узнать, какие задания предстоит выполнять в течение семестра. Поэтому время сдачи заданий может быть не таким жестким, как при традиционной форме обучения, но все задания нужно обязательно сдать к концу так называемого «спринта» или всего курса.

Немаловажная роль при организации дистанционных занятий отводится такому инструменту как видеоконференции. Сопоставление данных, опубликованных в источниках [7, 8] по наиболее популярным программным средствам для данного вида коммуникаций, приведено в Таблице 4.

Таблица 4 – Сравнение программного обеспечения для проведения видеоконференций

Наименование	Zoom	Skype	Discord	BigBlueButton
Общий чат	До 500 участников	+	+	+
Демонстрация экрана	+	+	До 25 человек в бесплатной версии	+
Количество участников в конференции	1000	50	50	До 300
Передача файлов	+	+	До 100МБ	-
Качество видео, р.	720р	1080р	1080р	720р
Ограничение бесплатной версии	40 минут записи, 40 минут групповых конференций, до 100 участников	Отсутствие SkypeIn, SkypeOut	Видео до 720р	15 мин. длительность вебинара До 5 участников в комнате
Стоимость платной версии	От 15$	Поминутная тарификация SkypeIn, SkypeOut	5$	2590р при установке на свой сервер
Дополнительные функции	Подмена фона	Субтитры	Создание каналов, их стилизация и оформление.	Изменение текста и элементов главной страницы

Из данных Таблицы 4 можно сделать вывод, что наилучшим вариантом для проведения потоковых лекционных занятий является Zoom как вмещающий наибольшее количество участников одновременно. Однако в бесплатной версии длительность такой конференции ограничена 40 минутами. Также недостатком Zoom является его недостаточная защищенность. Множество записей конференций находится в открытом доступе, поэтому, если запись онлайн-занятия не должна попасть в интернет, то лучше выбрать другую платформу для его проведения. Для практических (например, компьютерных лабораторных)

и семинарских занятий более подходит Discord, поскольку весь основной функционал реализуется при подключении до 25-30 человек. Данная платформа не имеет ограничений по времени конференции, есть лишь лимит по числу просмотров демонстрации экрана. При небольшой загруженности сети также приемлемым и удобным вариантом является BigBlueButton. Однако больших нагрузок BigBlueButton и Skype не выдерживают.

В каждом из названных сервисов есть встроенный чат, функционал которого примерно одинаков при наличии некоторых особенностей. Так, в Discord чат не привязан к конференции, и есть возможность создавать множество чатов для канала. Это удобно, поскольку можно разделить чаты для разных групп пользователей. В BigBlueButton есть вкладка заметок, которые доступны для написания любым подключенным пользователем, что несколько упрощает взаимодействие участников учебного занятия. В Zoom чат функционален, однако при демонстрации экрана его не сразу можно найти.

Заключение

В работе определен ряд достоинств и недостатков дистанционного учебного процесса как с точки зрения преподавателя, так и с точки зрения студента. Рассмотрена LMS-платформа Moodle, обращено внимание на ее недостатки, относящиеся, прежде всего, к удобству применения и к пользовательскому интерфейсу. В качестве альтернативы указан метод Scrum, который может помочь организовать удаленный учебный процесс более эффективно.

Рассмотрен ряд платформ, применяемых для проведения занятий в режиме видеоконференций: Zoom, Discord, BigBlueButton, Skype. Проведенное сопоставление позволило сделать вывод, что Zoom более предпочтительно использовать для лекционных занятий, а Discord или BigBlueButton для проведения практических и семинарских занятий. В то же время преподавателю должна быть предоставлена возможность выбора той или иной платформы в зависимости от собственных требований к программному обеспечению, используемому для организации и проведения занятий.

Результаты работы могут быть полезны при решении вопроса о выборе преподавателем программных средств для реализации режима удаленного обучения.

Список литературы

1. НИУ ВШЭ. Официальный сайт. Режим доступа: https://www.hse.ru/(дата обращения 29.04.2021).
2. Категория Agile. Режим доступа: habr.com/ru/company/hygger/blog/351048/ (дата обращения 29.04.2021).
3. Сазерленд Д. Scrum. Революционный метод управления проектами. М.: Изд-во ООО «Манн, Иванов и Фербер». 2016. 186 с.
4. Что такое Agile и Scrum и как с ним работать. Режим доступа: https://biz.mann-ivanov-ferber.ru/2019/09/09/chto-takoe-agile-i-scrum-i-kak-s-nimi-rabotat/(дата обращения 29.04.2021).
5. Инструкция по написанию CJM. Режим доступа: https://dkapaev.medium.com/%D0%BA%D1%80%D0%B0%D1%82%D0%BA%D0%B8%D0%B9-%D0%B3%D0%B0%D0%B9%D0%B4-%D0%BF%D0%BE-cjm-19667b50fd7a (дата обращения 29.04.2021).
6. Moodle. Официальный сайт. Режим доступа: https://moodle.org/(дата обращения 29.04.2021).
7. Discord. Официальный сайт. Режим доступа: https://discord.com/(дата обращения 29.04.2021).
8. Сравнение платформ для видеоконференций. Режим доступа: https://habr.com/ru/company/leader-id/blog/495094/ (дата обращения 29.04.2021).

References

1. HSE. Official website. Access mode: https://www.hse.ru/(accessed 29.04.2021).
2. Agile category. Access mode: habr.com/ru/company/hygger/blog/351048 / (date of issue 29.04.2021).
3. Sutherland D. Scrum. The revolutionary method of project management. M.: Publishing house of LLC "Mann, Ivanov and Ferber". 2016. p.186
4. What is Agile and Scrum and how to work with it. Access mode: https://biz.mann-ivanov-ferber.ru/2019/09/09/chto-takoe-agile-i-scrum-i-kak-s-nimi-rabotat / (accessed 29.04.2021).
5. Instructions for writing CJM. Access mode: https://dkapaev.medium.com/%D0%BA%D1%80%D0%B0%D1%82%D0%BA%D0%B8%D0%B9-%D0%B3%D0%B0%D0%B9%D0%B4-%D0%BF%D0%BE-cjm-19667b50fd7a (accessed 29.04.2021).
6. Moodle. Official website. Access mode: https://moodle.org / (accessed 29.04.2021).
7. Discord. Official website. Access mode: https://discord.com / (accessed 29.04.2021).
8. Comparison of video conferencing platforms. Access mode: https://habr.com/ru/company/leader-

Хрусталева М.С., Павлов А.В. Безопасность крупных энергосистем // Международный журнал информационных технологий и энергоэффективности. – 2023. – Т. 8 № 8(34) с. 60–63

Международный журнал информационных технологий и энергоэффективности

Сайт журнала:
http://www.openaccessscience.ru/index.php/ijcse/

УДК 621.311

БЕЗОПАСНОСТЬ КРУПНЫХ ЭНЕРГОСИСТЕМ

Хрусталева М.С., [1]Павлов А.В.
ФГБОУ ВО "ТВЕРСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ", Тверь, Россия (170026, Тверская область, город Тверь, наб. Афанасия Никитина, д.22), e-mail: [1] sokolhawk98@gmail.com

В статье рассматривается доступность динамической электросети. Теория объясняется на примере двухзонной энергосистемы, представленной агрегированной моделью колебаний. Количество отказов является критерием для принятия решений по управлению резервированием, повышающим доступность системы при возникновении сбоев из-за потери оборудования. Критерий позволяет включать формальные показатели неопределенностей, связанных с диагностикой неисправностей в режиме реального времени, а также формальные показатели эффективности контроля. Также исследовано влияние наличия несущей конструкции современной электросети на доступность самой электросети, с акцентом на сеть единиц измерения. Акцент сделан на признании растущей потребности в диагностике и контроле в режиме реального времени в современной энергосистеме.

Ключевые слова: Энергосистема, безопасность энергосистемы, надежность энергосистемы, отключение электроэнергии, динамическая безопасность, устойчивость энергосистемы.

SECURITY OF LARGE POWER SYSTEMS

Khrustaleva M.S., [1]Pavlov A.V.
TVER STATE TECHNICAL UNIVERSITY, Tver, Russia (170026, Tver region, Tver city, Afanasiya Nikitin embankment, 22), e-mail: [1]sokolhawk98@gmail.com

The article discusses the number of failures, how this affects the availability of a dynamic power grid. The theory is explained on the example of a two-zone power system represented by an aggregated oscillation model. The number of failures is a criterion for making redundancy management decisions, increasing the availability of the system in the event of failures due to loss of equipment. The criterion makes it possible to include formal indicators of uncertainties associated with real-time fault diagnosis, as well as formal indicators of control efficiency. The influence of the presence of the supporting structure of a modern electrical network on the availability of the electrical network itself is also investigated, with an emphasis on the network of units of measurement. Emphasis is placed on the recognition of the growing need for real-time diagnostics and control in the modern power system.

Keywords: Power system, power system security, power system reliability, power outages, dynamic security, power system stability.

Реструктуризация энергосистемы [1] подразумевает более высокую степень взаимосвязанности, в том числе использование возобновляемых источников энергии, и большую зависимость от информационных технологий в несущей структуре сети, где, среди многих компонентов, размещаются новые устройства управления и контроля. Сложность, неопределенность и уязвимость, сопровождающие новые возможности, усиливают потребность в формальном исследовании доступности, которое также должно учитывать

доступность растущей поддерживающей структуры и учитывать последствия решений, принимаемых в режиме реального времени.

Несмотря на замечательные достижения последних нескольких десятилетий в усилиях по повышению безопасности и надежности электрических сетей, только за последний 5 лет (2018-2023 г.г.) произошло семь крупных отключения электроэнергии в Мире [2]:

1. Венесуэла – 7 марта 23 июля 2019 года, пострадавших 30 млн. человек;
2. Аргентина, Парагвай, Уругвай – 16 июня 2019 года, пострадавших 48 млн. человек;
3. Индонезия – 4-5 августа 2019 года, пострадавших 120 млн. человек;
4. Шри-Ланка – 17 августа 2020 года, пострадавших 21 млн. человек.
5. Пакистан – 9 января 2021 года, пострадавших 200 млн. человек, что составляет ≈ 80% населения страны;
6. Бангладеш – 4 октября 2022 года, пострадавших 140 млн. человек, что составляет ≈ 80% населения страны;
7. Пакистан – 23 января 2023 г, пострадавших 230 млн. человек, что составляет ≈ 99% населения страны.

Автор The Unruly Power Grid [3] процитировал некоторые передовые исследования, которые предполагают, что крупные отключения электроэнергии неизбежны. Одна теория связывает такие события с недостаточным запасом устойчивости, поскольку преобладает стремление к максимальной отдаче от инвестиций в энергосистему, а другая — со сложностью все более взаимосвязанных систем, что находит поддержку в устоявшейся теории больших отклонений в многомерных рисках [4]. Обе теории опираются на то, что стоимость создания энергосистем, делающих редкие события еще более редкими, слишком высока, а сложность общепризнанно снижает надежность.

При анализе надежности энергосистемы для каждой из систем генерации, передачи и распределения использовалось множество отдельных показателей [5]. Кроме того, понятие безопасности использовалось для обозначения способности электроэнергетической системы выдерживать крупные изменения, например, из-за непредвиденной потери объектов системы [6]. Вероятностная структура использовалась для обеспечения унифицированной обработки стационарной устойчивости и динамической безопасности. Некоторые исследования включают гибкие системы передачи переменного тока (FACTS) в анализ надежности. Однако использование нескольких индексов и отдельного понятия безопасности указывает на то, что связь между переходным поведением, вызванным сбоями в динамической системе, и надежностью системы, на которую, по-видимому, влияет ее безопасность, не установлена должным образом. Эта связь полностью отсутствует, когда для управления резервированием оборудования используется неопределенная информация в реальном времени.

Первый тип неисправности связан с повышением безопасности при эксплуатации связано с сетевым оборудованием, таким как генераторные установки, трансформаторы, линии электропередачи и нагрузки. Конечная цель состоит в своевременном устранении этого типа отказов, насколько это возможно, за счет эксплуатации и управления резервированием оборудования, несмотря на наличие отказов второго типа.

Второй тип неисправностей связан с компонентами, находящимися в несущей конструкции сети, такими как устройства управления, обработки и измерения. Растущее присутствие новых мощных и быстрых устройств в системах и высоковольтных системах

передачи постоянного тока, а также векторных измерительных устройствах значительно расширило возможности использования и управления резервированием в энергосистеме.

Поскольку всегда ожидается восстановление работы сети, устойчивая доступность, а не надежность, принимается в качестве меры производительности сети. Доступность в устойчивом состоянии — это примерно доля времени в долгосрочной перспективе, в течение которой система работает удовлетворительно. Степень удовлетворенности может быть определена на нескольких уровнях и включать экономические факторы. В рамках стохастической модели с дискретными состояниями доступность может быть вычислена как сумма вероятностей состояний для состояний, которые классифицируются как представляющие оперативную сетку с заданным порогом производительности.

Восстановление системы зависит от того, настроена ли сеть на допустимое количество потерь оборудования. Если это так, то защитные устройства, такие как реле и автоматические выключатели [7], должны удалить неисправное оборудование до критического времени отключения, после которого межзональная синхронизация теряется.

Однако следует отметить, что помехи во время неисправности могут привести к срабатыванию защитных устройств в неповрежденном оборудовании и, таким образом, к каскадным событиям, когда защитные устройства не могут различить причины колебаний напряжения и тока. Диагностическая поддержка в сети недостаточна для изменения предварительно установленных протоколов повторного включения в режиме реального времени. Хотя ожидается, что новые схемы, которые возвращают измеренную или оцененную информацию по обширной области модернизируемой сети, будут способны выполнять более сложное управление резервированием в режиме реального времени, риски, связанные с неопределенностями в измеренной или оцененной информации, должны быть официально включены в решение. процесс.

Надлежащей мерой производительности энергосистемы является ее способность возвращаться к желаемому равновесию. Рассчитанное покрытие отказов для двухзонной системы фиксирует эффект отсроченного контроля или действий по управлению, эффект неопределенных результатов диагностики и эффект управляемости.

Тенденция к получению и обработке информации в режиме реального времени при работе современной энергосистемы делает все более важным использование измерительных устройств. Все схемы размещения носят детерминированный характер, и связь между доступностью данных измерений и доступностью сетки не установлена.

При проектировании архитектуры резервирования необходимо смоделировать доступность данных измерений. Новая проблема проектирования структуры измерительной сети формулируется как задача оптимизации, в которой цены назначаются конфигурациям системы и применяются ограничения на доступность данных.

Список литературы

1. Chow J.H., Wu F.F., Momoh J.A. (ed). Applied mathematics for restructured electric power systems: optimization, control, and computational intelligence // Springer, USA. 2005. 233 p.
2. Список основных отключений электроэнергии: Википедия. [Электронный ресурс]. URL: https://translated.turbopages.org/proxy_u/en-ru.ru.77412d65-64dc9bc5-11cd7738-74722d776562/https/en.wikipedia.org/wiki/List_of_major_power_outages (дата обращения: 05.08.2023).

3. Fairley P. The unruly power grid // IEEE Spectrum. 2004. №41. PP 22–27.
4. Dembo A, Zeitouni O. Large deviations techniques and applications. Applications of mathematics // Springer-Verlag, Berlin. 1998. №38. 396 p.
5. Mountford J.D., Austria RR (1999) Keeping the lights on. IEEE Spectrum 36:34–39
6. Varaiya P., Wu F.F., Chen R. Direct methods for transient stability analysis of power systems: recent results // Proc IEEE. 1985. №73. 1703–1715
7. Anderson P.M. Power system protection // IEEE Press, New Jersey. 1998

References

1. Chow J.H., Wu F.F., Momoh J.A. (ed). Applied mathematics for restructured electric power systems: optimization, control, and computational intelligence // Springer, USA. 2005. 233 p.
2. List of major power outages: Wikipedia. [Online]. URL: https://translated.turbopages.org/proxy_u/en-ru.ru.77412d65-64dc9bc5-11cd7738-74722d776562/https/en.wikipedia.org/wiki/List_of_major_power_outages (дата обращения: 05.08.2023).
3. Fairley P. The unruly power grid // IEEE Spectrum. 2004. №41. PP 22–27.
4. Dembo A, Zeitouni O. Large deviations techniques and applications. Applications of mathematics // Springer-Verlag, Berlin. 1998. №38. 396 p.
5. Mountford J.D., Austria RR (1999) Keeping the lights on. IEEE Spectrum 36:34–39
6. Varaiya P., Wu F.F., Chen R. Direct methods for transient stability analysis of power systems: recent results // Proc IEEE. 1985. №73. 1703–1715
7. Anderson P.M. Power system protection // IEEE Press, New Jersey. 1998

Сорокин Ф.А. Определение основных параметров ионисторного накопителя энергии вагона метрополитена, работающего с электрическим приводом переменного тока // Международный журнал информационных технологий и энергоэффективности. – 2023. – Т. 8 № 8(34) с. 64–70

Международный журнал информационных технологий и энергоэффективности

Сайт журнала:
http://www.openaccessscience.ru/index.php/ijcse/

УДК 621.355

ОПРЕДЕЛЕНИЕ ОСНОВНЫХ ПАРАМЕТРОВ ИОНИСТОРНОГО НАКОПИТЕЛЯ ЭНЕРГИИ ВАГОНА МЕТРОПОЛИТЕНА, РАБОТАЮЩЕГО С ЭЛЕКТРИЧЕСКИМ ПРИВОДОМ ПЕРЕМЕННОГО ТОКА

Сорокин Ф.А.
ФГБОУ ВО "ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ПУТЕЙ СООБЩЕНИЯ ИМПЕРАТОРА АЛЕКСАНДРА I", Санкт-Петербург, Россия (190031, г.Санкт-Петербург, пр-кт Московский, Д.9), e-mail: filippovna1965@mail.ru

В данной работе предложены основные зависимости для расчета массогабаритных и энергетических характеристик ионисторного накопителя энергии, работающего с электрическим приводом переменного тока. Далее был произведен расчет для 8-ми вагонного поезда метро с приводом переменного тока, под общим коммерческим названием «Москва».
Актуальность этой работы заключается в том, что на данный момент отсутствуют решения расчета массогабаритных и энергетических характеристик ионисторного накопителя энергии, работающего с электрическим приводом переменного тока электропоезда метро.

Ключевые слова: Экономичный Тяговый преобразователь, емкостной накопитель энергии, ионистор, вагон метрополитена, энергия рекуперации, батарея.

DETERMINATION OF THE MAIN PARAMETERS OF THE IONISTOR ENERGY STORAGE OF THE METRO CAR WORKING WITH THE ELECTRIC DRIVE OF THE AC

Sorokin F.A.
Petersburg State University of Communications of Emperor Alexander I, St. Petersburg, Russia (190031, Saint-Petersburg, Moskovsky Ave., 9), e-mail: filippovna1965@mail.ru

In this paper, the main dependences for calculating the weight, size and energy characteristics of an ionistor energy storage device operating with an AC electric drive are proposed. Next, a calculation was made for an 8-car metro train with an AC drive, under the general commercial name "Moscow".
The relevance of this work lies in the fact that at the moment there are no solutions for calculating the weight, size and energy characteristics of an ionistor energy storage device operating with an electric AC drive of a metro electric train.

Keywords: Economical traction converter, capacitive energy storage, ionistor, subway car, energy recovery, battery.

1. Определение величины энергии рекуперативного торможения и потребной емкости ионисторного накопителя для электропоезда метро

В соответствии с [1], расчетным режимом движения для вагонов метрополитена является движение на прямом, горизонтальном участке пути с сухими рельсами при нормальном

напряжении в контактной сети и полезной нагрузки на расчетном перегоне длиной 1700 м. Для расчетов принимаются основные параметры для 8-ми вагонного поезда метро серии 81-765, 766 и 767 под общим коммерческим названием «Москва».

Для данного перегона энергия рекуперации поезда $A_{\text{р}}$, кВт · ч, составит:

$$A_{\text{р}} = 36{,}389 \text{ кВт} \cdot \text{ч}.$$

Так как в данной работе ведется разработка накопителя для одного моторного вагона, то далее определяется энергия рекуперации, приходящаяся на один моторный вагон $A_{\text{р1}}$, кВт · ч, определяется по выражению (2):

$$A_{\text{р1}} = \frac{A_{\text{р}}}{(n_{\text{гл}} + n_{\text{мот}})}, \quad (2)$$

где $n_{\text{гл}}$ и $n_{\text{мот}}$ - количество головных моторных и прицепных моторных вагонов соответственно, шт.

$$A_{\text{р1}} = \frac{36{,}389}{(2+4)} = 6{,}035 \text{ кВт} \cdot \text{ч}.$$

Выражение для определения емкости ионисторного накопителя для одного моторного вагона $\text{С}_{\text{ен}}$, Ф, по выражению (3):

$$\text{С}_{\text{ен}} = \frac{2 \cdot A_{\text{р1}}}{(U^2_{\text{ен}\,max} - U^2_{\text{ен}\,min})}. \quad (3)$$

где $U_{\text{ен}\,max}$ и $U_{\text{ен}\,min}$ – максимальное и минимальное соответственно напряжение ионисторного накопителя энергии.

Предварительно принимая $U_{\text{ен}\,max} = 1000$ В и $U_{\text{ен}\,min} = 500$ В:

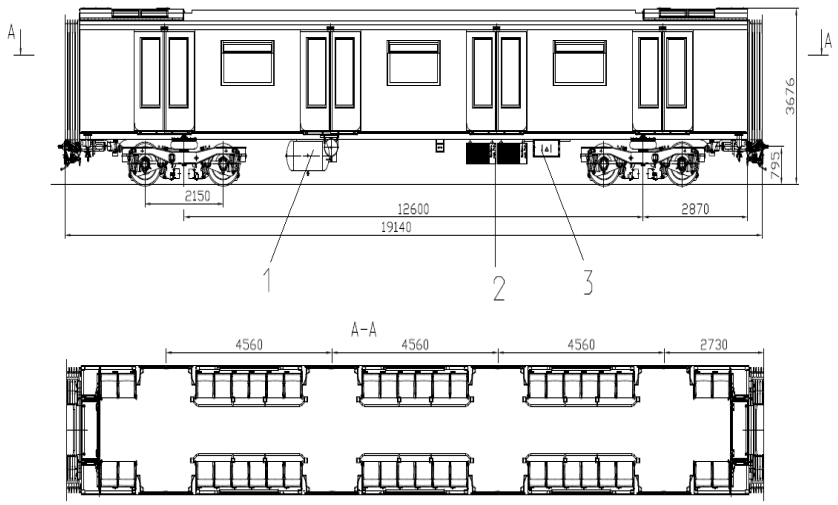
$$\text{С}_{\text{ен}} = \frac{2 \cdot 6{,}035 \cdot 1000 \cdot 3600}{(1000^2 - 500^2)} = 58{,}2 \text{ Ф}.$$

Таким образом, емкость накопителя составляет $\text{С}_{\text{ен}} = 58{,}2$ Ф. В следующем пункте предлагается произвести расчеты основных параметров ионисторного накопителя.

2. Определение габаритов и основных параметров ионисторного накопителя энергии

В виду отсутствия свободного места непосредственно на моторных вагонах модели 81-765 и 81-766, предлагается установка на свободное пространство между тележками под прицепным вагоном модели 81-767 (Рисунок 1).

При определении величины свободной площади под прицепным вагоном необходимо учитывать площади, занимаемые вспомогательным оборудованием (преобразователь кондиционера салона, воздушные резервуары и распределитель) [2].



1 – воздушные резервуары, 2 - преобразователь кондиционера салона и 3 – блок распределителей.

Рисунок 1 – Общий вид вагона модели 81-767

Источник: РЭ вагонов модели 81-765, 81-766 и 81-767

С учетом вышесказанного свободная площадь $S_{\text{ен}}$, которая может быть использована для монтажа ящиков с ионисторами, определяется по выражению (4), м^2:

$$S_{\text{ен}} = S_{\text{мп}} - S_{\text{вр}} - S_{\text{пк}} - S_{\text{бр}}, \quad (4)$$

где $S_{\text{мп}}$ – площадь межтележечного пространства, м^2;

$S_{\text{вр}}$ – площадь, воздушными резервуарами, м^2;

$S_{\text{пк}}$ – площадь, занимаемая ящиком с преобразователем кондиционеров салона, м^2;

$S_{\text{БР}}$ – площадь, занимаемая ящиком с блоком распределителей, м^2.

Для прицепного вагона модели 81-767 $S_{\text{ен}} \approx 20{,}5$ м^2, при этом максимальная длина накопителя $L_{\text{ен}\,max} \approx 7{,}6$ м и максимальная ширина накопителя $B_{\text{ен}\,max} \approx 2{,}7$ м.

Для сборки ионисторной батареи был выбран накопитель НСКБ-83-102 (рисунок 2). Его основные параметры представлены в Таблице 1.

Рисунок 2 – Общий вид накопителя суперконденсатоного НСКБ-83-1023

Источник: РЭ накопителя суперконденсатоного НСКБ-83-1023

Таблица 1 – Характеристики суперконденсатоного накопителя НСКБ-83-102

Диапазон рабочих напряжений, В	102-51
Максимальное напряжение, В	102
Минимальное напряжение, В	51
Емкость, Ф	83
Внутренне сопротивление, мОм	11 мОм
Рабочий диапазон температур, ˚С	-40…+65
Рабочий ресурс, циклы заряд-разряд	1000000
Максимальный постоянный ток (15/10˚С), А	128/208
Габариты, мм Высота Длина Ширина	 177 484 548
Масса, кг	32

Источник: РЭ накопителя суперконденсатоного НСКБ-83-1023

Далее определяется схема соединения ионисторных накопителей энергии.

В соответствии с [3] максимальное напряжение в звене постоянного тока инвертора будет достигаться в режиме электрического торможения и будет составлять 940 В. Далее определяется количество последовательно соединённых накопителей $N_{\text{ен пс}}$ батареи, шт., по выражению (5):

$$N_{\text{ен пс}} = \frac{U_{d\,max}}{U_{\text{н}}} = \frac{940}{102} \approx 10 \text{ шт.} \qquad (5)$$

При этом емкость группы из 10 последовательно соединённых ионисторных накопителей $\text{С}_{\text{ен пс}}$, Ф, составит по выражению (6):

$$\text{С}_{\text{ен пс}} = \frac{\text{С}_{\text{ен 1}}}{N_{\text{ен пс}}} = \frac{83}{10} = 8{,}3 \text{ Ф.} \qquad (6)$$

В таком случае, сопротивление группы из 10 последовательно соединённых ионисторных накопителей $R_{\text{ен пс}}$, мОм, по выражению (7) составит:

$$R_{\text{ен пс}} = N_{\text{ен пс}} \cdot R_{\text{ен 1}} = 10 \cdot 11 = 110 \text{ мОм.} \qquad (7)$$

Далее определяется количество параллельных групп накопителей $N_{\text{ен пр}}$ из условия набора емкости в 58,2 Ф, по выражению (8):

$$N_{\text{ен пр}} = \frac{\text{С}_{\text{ен}}}{\text{С}_{\text{ен пс}}} = \frac{58{,}2}{8{,}3} \approx 7 \text{ шт.} \qquad (8)$$

Также нужно провести проверку на максимальный длительный ток батареи, А, по выражению (9):

$$I_{\text{ен 1}} \cdot N_{\text{ен пр}} \geq I_{d\,max};\ \text{А}, \qquad (9)$$

$$170 \cdot 7 \geq 1102,\ \text{А}$$

$1190 \geq 1102, \text{А}$ – условие выполняется.

Определяется фактическая емкость накопителя $C_{\text{ен пс}}$, Ф, определяется по выражению (10):

$$C_{\text{ен пс}} = C_{\text{ен пс}} \cdot N_{\text{ен пр}} = 8{,}3 \cdot 7 = 58{,}1 \text{ Ф}. \quad (10)$$

Таким образом, для обеспечения общей емкости ионисторной батареи, в 58,1 Ф, предлагается соединение 7 параллельных групп накопителей по 10 штук в каждой.

В таком случае, сопротивление ионисторной батареи $R_{\text{ен пс}}$, мОм, по выражению (11), составит:

$$R_{\text{ен пс}} = \frac{R_{\text{ен пс}}}{N_{\text{ен пр}}} = \frac{110}{7} = 15{,}71 \text{ мОм}. \quad (11)$$

Так как аккумуляторы каждого моторного вагона будут располагаться на прицепных вагонах, а для 8-ми вагонного состава их число составляет 2 шт, поэтому на каждом из двух прицепных вагонов будет располагаться батарея, состоящая из 3 аккумуляторов.

В соответствии с этим, общее количество накопителей для одного прицепного вагона $N_{\text{ен 1в}}$, шт, по формуле (12) составит:

$$N_{\text{ен 1в}} = N_{\text{ен пс}} \cdot N_{\text{ен пр}} \cdot 3 = 10 \cdot 7 \cdot 3 = 210 \text{ шт}. \quad (12)$$

Тогда, масса батареи одного прицепного вагона $m_{\text{ен 1в}}$, кг, по выражению (13), составит:

$$m_{\text{ен 1в}} = N_{\text{ен 1в}} \cdot m_{\text{ен 1}} \cdot 3 = 70 \cdot 32 \cdot 3 = 6720 \text{ кг}. \quad (13)$$

Для наибольшей энергоемкости на единицу площади предлагается располагать каждый накопитель лицевой панелью вверх, ориентируя сторону корпуса накопителя длинной 548 мм в сторону тележек вагона.

Далее определяется максимальное количество последовательно сориентированных накопителей по условиям ширины подвагонного пространства $N_{\text{ен пс}}$, шт, по выражению (14):

$$N_{\text{ен пс}} = \frac{L_{\text{ен } max}}{L_{\text{ен 1}}} = \frac{7648}{484} \approx 15 \text{ шт}. \quad (14)$$

Затем определяются необходимое количество параллельно сориентированных накопителей $N_{\text{ен пр}}$, шт, по выражению (15):

$$N_{\text{ен пр}} = \frac{N_{\text{ен 1в}}}{N_{\text{ен пр}}} = \frac{210}{15} = 14 \text{ шт}. \quad (15)$$

В таком случае, ширина батареи $B_{\text{ен}}$, мм, определяется выражением (16):

$$B_{\text{ен}} = B_{\text{ен 1}} \cdot N_{\text{ен пр}} = 177 \cdot 14 = 2478 \text{ мм}. \quad (16)$$

В таком случае, длина батареи $L_{\text{ен}}$, мм, определяется выражением (17):

$$L_{\text{ен}} = L_{\text{ен 1}} \cdot N_{\text{ен пс}} = 484 \cdot 15 = 7260 \text{ мм}. \quad (17)$$

Высота батареи $H_{\text{ен}}$, мм, определяется выражением (18):

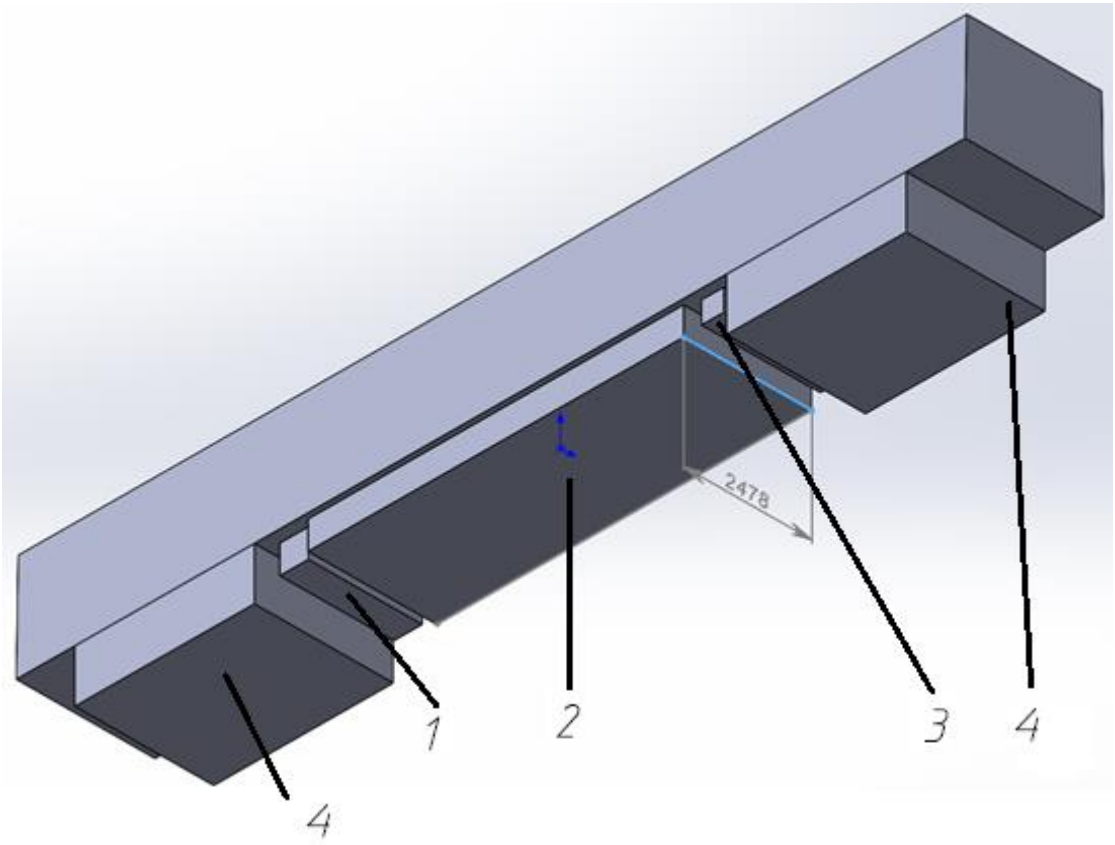
$$H_{\text{ен}} = H_{\text{ен 1}} = 548 \text{ мм}. \quad (18)$$

Все рассчитанные параметры были сведены в таблицу 2. Также вариант расположения ионисторного накопителя в подвагонном пространстве вагона модели 81-767 представлен на Рисунке 3.

Таблица 2 –Характеристики суперконденсатоной батареи

Диапазон рабочих напряжений, В	1020-510
Максимальное напряжение, В	1020
Минимальное напряжение, В	510
Количество вагонов, запутываемых от батареи, вагонов	3
Емкость, Ф	$58{,}1 \cdot 3$
Внутренне сопротивление, мОм	15,7 мОм
Рабочий диапазон температур,°С	-40…+65
Рабочий ресурс, циклы заряд-разряд	1000000
Максимальный постоянный ток (15/10°С), А	896/1456
Габариты, мм: Высота Длина Ширина	 548 7260 2478
Масса, кг	6720

Источник: Выше представленные расчеты



1 - воздушные резервуары, 2-ионисторная батарея, 3 - преобразователь кондиционера салона и 4 - тележки.

Рисунок 3 – Общий вид ионисторной батареи с указанием ширины батареи

Источник: Авторский материал

Список литературы

1. ГОСТ Р 50850 – 96 Вагоны метрополитена. Общие технические требования.
2. Васильев, В.А. Повышение энергетической эффективности электропоездов постоянного тока [Рукопись]: дис.канд. техн наук: 05.22.07: защищена 06.03.12 / Васильев Виталий Алексеевич. – Санкт-Петербург, 2012. – 125 с.
3. Комплект электрооборудования асинхронного тягового привода вагонов метрополитена КАТП-3. Руководство по эксплуатации. – М.: ОАО «Метровагонмаш». 2016. – 236 с.

References

1. GOST R 50850 – 96 Subway cars. General technical requirements.
2. Vasiliev, V.A. Improving the energy efficiency of DC electric trains [Manuscript]: dis.Candidate of Technical Sciences: 05.22.07: protected 06.03.12 / Vasiliev Vitaly Alekseevich. – St. Petersburg, 2012. – p.125
3. A set of electrical equipment for asynchronous traction drive of subway cars KATP-3. Operating manual. – M.: JSC "Metrovagonmash". 2016. – p.236