Advances in Artificial Intelligence and Machine Learning; Research 4 (4) 3161-3172 Received 18-10-2024; Accepted 22-12-2024; Published 29-12-2024

A Script Language Creation for Event Visualization and Data Storage in an Intelligent Video Surveillance System

Alexey Subbotin alesu1543@gmail.com
Saint Petersburg Electrotechnical University,
"LETI" (ETU)
St. Petersburg,
Russia.

Radhakrishnan Delhibabu rdelhibabu@vit.ac.in
School of Computer Science and Engineering,
Vellore Institute of Technology, Vellore
India.

Nataly Zhukova nazhukova@mail.ru
Saint-Petersburg Federal Research
Centre of the Russian Academy of Sciences
St. Petersburg,
Russia.

Corresponding Author: Radhakrishnan Delhibabu

Abstract

The development of a macro-based script language for storing data in a database base and visualizing events in an intelligent video surveillance system is discussed. An overview of the built-in script languages used in systems is presented. A comparative analysis of general-purpose script languages is given. A problem with the speed of data access has been identified. Our script programming language based on macros and templates for visualizing events and storing data is proposed as a solution to the problem. The proposed approach made it possible to reduce data access time by 19.34% relative to the fastest PHP script language and by 6.32 times relative to the slowest Ruby language.

Keywords: Script language, Script language creation, Event visualization, Data storage, Intelligent video surveillance system.

1. INTRODUCTION

Many programs use embedded programming languages, which are the implementation of high-level languages. For example, Docsvision electronic document management system [1], applies its own C#-based script language. SAP Business One Enterprise Management Systems (ERP) [2], apply their own programming language ABAP. The domestic 1C system [3], has its own script language in the 1C Configurator, based not on English, but on Russian. The system of automatic collection and metering of electricity ”Pyramid 2.0” [4], uses its own language based on Silverlight and C#, not just SQL. All script languages are executed at the kernel level (executable *.exe file) and in plug-ins (based on *.dll libraries). The script languages in the program interact with objects to expand functionality: create additional reports, tables, forms, and interact with the database. In some cases, template functions are created that can be copied, edited, expanding the basic capabilities of the program. As a rule, the interpreter of the script language is already in RAM and scripts are executed much faster than when running as a separate program on disk. Speed (code execution speed) and functionality are the two main indicators of an embedded programming language for a specific program. When creating your own programming language, it is necessary to take into account not only the functionality, which is the presence of a huge number of plug-ins as in popular programming languages, but also interaction with the main program interface (API) through a collection of functions with detailed documentation [5, 6, 7]. In the followinf cases usage of popular programming languages is not reasonable::

1. High security is required. In script language only a strictly defined set of functions (API) is available to the user.
2. High speed of interaction. When interacting with ready-made program modules: forms, reports, blocks on the website, etc., the script language works much faster.

The expansion of the program’s functionality due to the script language has been known for a long time. The first script language provided the function of reading the configuration file, where the settings were represented in blocks. For example, the line: Name ”News from around the world”. Here, the parameter is Name, and the value is ”News from around the world”, which is separated by a space. The example is taken from the RSS [8], news stream reader from the ”full.cfg” file. The syntax for setting parameters can be extended by condition operators, which already resemble a scripting programming language. You can perform the configuration with the command: ”rss2fido.exe -c full.cfg”. Terminal programs that were popular in the 90s are widely known. For example, the well-known Tornado terminal program [9], has its own script language, which consists of a catalog of procedures resembling the high-level Pascal language. The conditions and procedures are filled in a text file and extend the basic functionality of Tornado.

2. OVERVIEW OF SCRIPTING PROGRAMMING LANGUAGES (BACKGROUND)

All script languages have general features, which include:

1. Creating functions with parameters or procedures;
2. Data types defined by the architecture of Intel and AMD family processors based on the IBM x86 assembler specification;

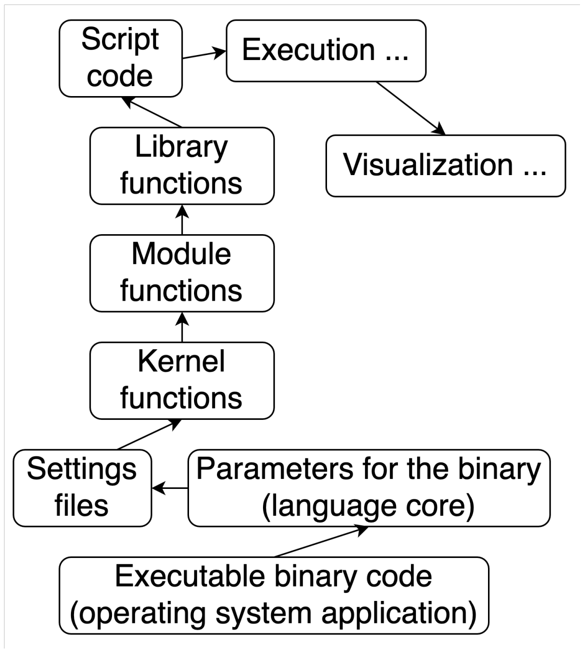


Figure 1: A typical execution scheme of an application programming language for a wide range of tasks

3. Condition or branching operators (if, for, while, until, switch), which are the implementation of the basic conditions and operations defined in a high-level language (C++, Java, Delphi) and that are interpreted in the logic of the kernel (a script language handler program);
4. Conditional start of script execution (beginning of file, main function, etc.);
5. Object-oriented script languages have objects (or classes) that can have methods (or functions) and fields (or variables).

With knowledge of the general features of all scripting programming languages, it is possible to create your own universal script language that will differ in syntax: other condition operators, function and procedure designations, classes, object creation and variable initialization. On the one hand, it is not difficult to make an embedded language: the script file is read, conditional expressions are parsed and logic in the program is executed. But only if it is a narrowly limited language (application level of the program), and not a general-purpose script language. The development of a general-purpose script language for solving various tasks, such as: Python, Ruby, Elixir, Go, Dart, R, Perl, Rust and etc., require, in addition to the unique syntax of the language itself, a significant set of additional modules for solving various tasks: statistics processing, working with databases, creating web-servers, machine learning and neural networks [10, 11, 12]. All these and many other modules require significant labor efforts to create such functionality.

The operation of any scripting programming language implies the following actions in the operating system (FIGURE 1): 1) reading parameters; 2) parsing settings from files; 3) loading modules; 4) reading libraries; 5) executing a script with visualization.

In fact, any script language is an application program for an operating system with many abstractions, that depend on the requirements of users. The script language does not involve compilation and will always work slower, it is a single program with any logic for all occasions, and the speed of development for a particular task is determined by the availability of ready-made modules, components and libraries. In some cases, one task can be solved for years, or it can be solved in weeks. The lack of the necessary library for machine learning in Ruby could stretch software development for years; the ready-made TensorFlow library for Python allows reduce code development to several weeks, along with documentation for the user of the intelligent video surveillance system [13, 14, 15].

PHP [16], is a popular programming language for creating websites. XAMPP 8.2.4 [17], standard package includes 41 modules for PHP, and the current version of PHP 8.3.8 contains more than 152 modules. In addition to creating websites, PHP has a large number of modules to expand the basic set of functions that are available in any programming language. These modules allow:

1. Image processing (module: php-gd),
2. Calculate numbers with any precision (module: GMP);
3. Work with FTP (directive: extension=php_ftp.dll);
4. Work with streams using the «stream_context_create()» function;
5. Exchange data over the SOAP protocol (the module is installed with the command: sudo apt-get install php7.1-soap);
6. Exchange data via SSH2 (the libssh2 module is installed via the website [18]).

Python [19], is the most popular script language for solving any application tasks and is supported by development environments (NetBeans, PyCharm, VisualStudio, IntelliJIdea, Eclipse). The simple syntax and a large number of libraries (over 137 thousands) in Python version 3.12.4 allow you to solve a huge number of tasks without compiling the bytecode (as in GCC, GPP, BilderC++, Delphi, FreePascal, etc. compilers), however, compiling a script in Python is acceptable as an option. Popular libraries for Python are frameworks for creating websites (library: Flask), working with HTTP(s) requests (library: Requests), building deep machine learning models (library: TensorFlow), data analysis (library: Pandas), working with matrices and mathematical operations (library: NumPy), and a huge number of other libraries that simplify the routine work of a programmer.

Perl [20], is the oldest programming language that was invented for the preparation of technical documentation. A distinctive feature of the language is the support of regular expressions at the core level, not due to the plug-in. Perl is slower than PHP, so it is not often used to create websites. Popular modules are: working with text (Text::CSV), working with JSON, XML, XLSX file formats and working with network objects (Net::SMTP, Net::Telnet, HTTP::Response, IO::Socket::SSL).

Ruby [21], has the most concise syntax of all script languages and was conceived by the developers as an improved version of Perl that provides all the advantages of Perl, but it is easier to develop scripts using Ruby (as in PHP), in Ruby actions are methods of objects and can return values. The redundancy of abstraction greatly slows down execution, but this is a requirement of Ruby developers. The most popular Ruby libraries are: Pagination (Pagy), a framework for creating websites (Ruby on Rails) with support of MVC templates, reading and writing Excel (Spreadsheet), working with images (Rmagick), reading XML (Nokogiri) and JSON, the Jammit compression library and many other libraries of the Ruby language, which allow to create not only a good website, but much more, competing with Perl.

Other script languages (Zig, Clojure, Elixir, Lisp, Scala, Go, OCaml, Flow, Groovy, Haskell, Apex, SAS, Nim, Raku, APL, Crystal, Julia) are less popular due to the lack of additional libraries, without which you can not do much, although they have detailed documentation on the collection of existing functions.

3. PROBLEM STATEMENT AND ISSUE DEFINITION

When solving general tasks it is hard to compete with a general-purpose script language for a number of reasons:

1. A very time-consuming process of recreating all modules (libraries, components);
2. Developers writing programs in script languages are getting used to a certain syntax;
3. Existing script languages work faster (for execution) and allow fewer kernel function errors due to the large number of users who simultaneously test the language.

However, in relation to a specific task (formation of forms, reports, tables) and interaction with a specific environment (the main program), some script languages work faster and more efficiently (1C, ABAP, etc.) than external scripts. Based on this statement, it can be assumed that a macro-based script language will work faster for visualizing events and storing data in the intelligent video surveillance systems. In addition, such a script language will be safe from the developer's point of view (deletion, modification of data), since it will have a strictly defined API. If we consider the issue of creating a script language for the intelligent video surveillance system, then a number of problems may arise:

1. Real-time data processing efficiency. The video surveillance system generates large amounts of data that needs to be quickly analyzed and visualized. The script language should be optimized for high-performance processing of streaming data.
2. Integration with various data sources. In addition to video, the system can use sensor data, metadata, and information from external systems. The language should provide simple and flexible integration with heterogeneous sources.
3. The convenience of visualizing events. It is important to provide users with intuitive tools for visualizing events, trends, and for analytics. The language must support powerful graphics libraries and visual components.
4. Scalability and distribution. A video surveillance system can cover a variety of cameras and sensors located in different places. The language should provide scalability and the possibility of distributed data processing.

5. Security and access control. Since the system works with confidential data, it is important to implement reliable authentication, authorization and data protection mechanisms.
6. Extensibility and modularity. The language should allow the addition of new functions, libraries, and integration with external components as the system develops.

Solving these problems requires careful design of the language architecture, selection of appropriate technologies and libraries. It is also important to ensure the usability of the language for developers. In this research, the problem is posed only with regard to the visualization of events and data storage in the intelligent video surveillance systems. To solve the problem of the insufficiently high speed of visualization of events on the website and interaction with the database, it is necessary to develop a new storage method, a type of a database management system (DBMS) and an optimized macro language, which can be easily used by the developers of intelligent video surveillance systems.

4. THE WAY TO SOLVE THE PROBLEM

The solution of the defined problem is to develop your own scripting programming language, created using GCC/GPP compilers based on C/C++ languages. A simple binary file will be executed much faster because it does not have additional configuration modules, libraries and other components that are read in other application programming languages from the directory with the language core, such as PHP, Perl, Ruby, etc. The script language in the intelligent video surveillance system will perform strictly defined functions from the core, so reading and writing to the database will be much faster, as well as creating HTML/CSS pages from macro templates for visualizing events that were determined using the Python pattern recognition system and the TensorFlow library [22, 23, 24].

Abandoning the additional modules of the language core (FIGURE 2) which are: settings, libraries, etc., you can immediately proceed to reading templates and working with the NoSQL database, which greatly reduces the execution time of complex code and increases the speed of event visualization and database work in the intelligent video surveillance systems.

The script language, which is conventionally called SubScript, is quite simple. As a parameter of the kernel ”sub2.1a.exe” the template is transmitted in HTML5 format, for example: ”sub2.1a.exe -t select.html”. There are non-standard elements in the DOM markup, which are conditionally called SubScript macros. For example, print the number of columns in the table: ”@cols”; add a table to the template with a limit of no more than 15 first columns: ”@tr<15”. Condition operators are applied, for example: ”@if @cols<25 to @tr”. Such a condition in the template ”select.html” allows you to display a table only if there are fewer than 25 columns. The logic of the SubScript 2.1a script language is limited only to event visualization and data storage. Therefore, the SubScript language is very simple and secure. For example, the function of writing to the DBMS table for logging is: ”@log_event: danger @text: shoe stuck in escalator”. SubScript is a very simple and understandable language that is clear to any novice programmer, the language makes it easy to add functionality for visualizing and storing event data in the intelligent video surveillance systems. It is supposed to use MongoDB [25] DBMS in NoSQL format, since NoSQL works faster with big data of the intelligent video surveillance systems compared to relational databases that have entity restrictions. The JSON format ensures compatibility with the Node software.js (JavaScript) used on the backend. Storage takes place in the key-value format, which works faster with big data of the

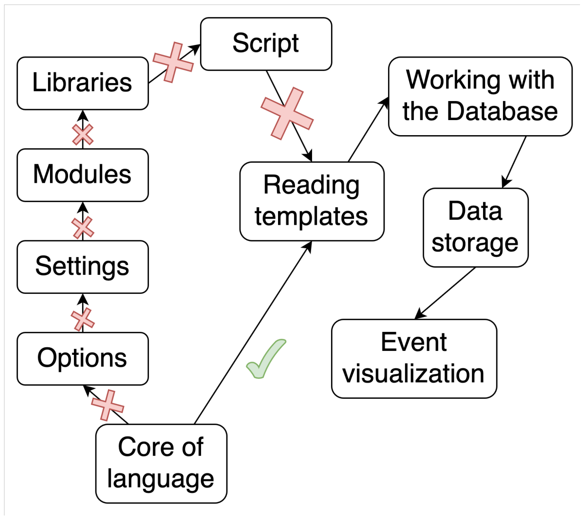


Figure 2: A way to solve the indicated problem

intelligent video surveillance systems than the SQL format (MySQL, PostgreSQL, Microsoft SQL Server, MariaDB, Oracle Database).

5. USAGE EXAMPLE

The authors of the research developed a software for visualizing events and storing data of the intelligent video surveillance system using a new script language based on macros and HTML5 templates with CSS 3.2 styles. Conditionally, the new language with macros was called SubScript, and was integrated into the core of the intelligent surveillance system.

On FIGURE 3, the main screen of the developed application for the smartphones is presented. Under the ”Event” label in the center of the screen, a certain ”Boot stuck” event is displayed on the screen of iPhone 15 Pro. The event can be confirmed with the ”OK” button or defined as an error of event recognition (the ”Error” button). The iOS application for the operator of the intelligent video surveillance system was developed in XCode in the Swift language. In this particular case, the ”Escalator” object is used, its 3D model is presented on the left on the smartphone screen. It is possible to rotate the object 360°, using the arrows on the right and left. Using the ”Menu” button on the main screen of the application it is possible to get access to additional settings. Rotation of the model is allowed in any direction. A red marker and the inscription ”Danger!” indicate the place of the event on the object. The application ”dtEvent-0.21a.ipa” is launched from the smartphone’s file system in developer mode, it sets permissions, tests devices, connects to the intelligent video surveillance system. Data is exchanged over the HTTP(s) protocol. After establishing a permanent connection and configuring the TCP/IP route, data is exchanged and information about the event is received in milliseconds. Information about the localization of detected event is transmitted as a coordinate in the three-dimensional XYZ space in JSON format: {” id”: 10093, ”name”: ”Danger”, ”level”: 3, ” time”: 11:27, ” xyz”: [652, 139, 3197]}. The coordinates are transmitted in millimeters at a scale of 1:52, which is sufficient for accurate positioning.

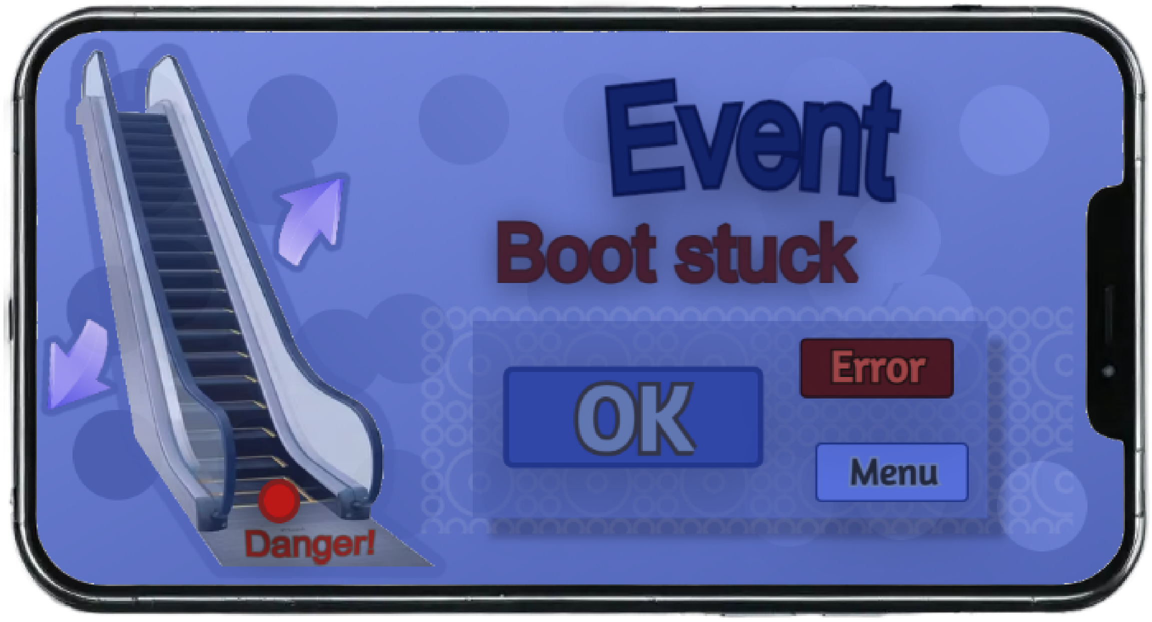
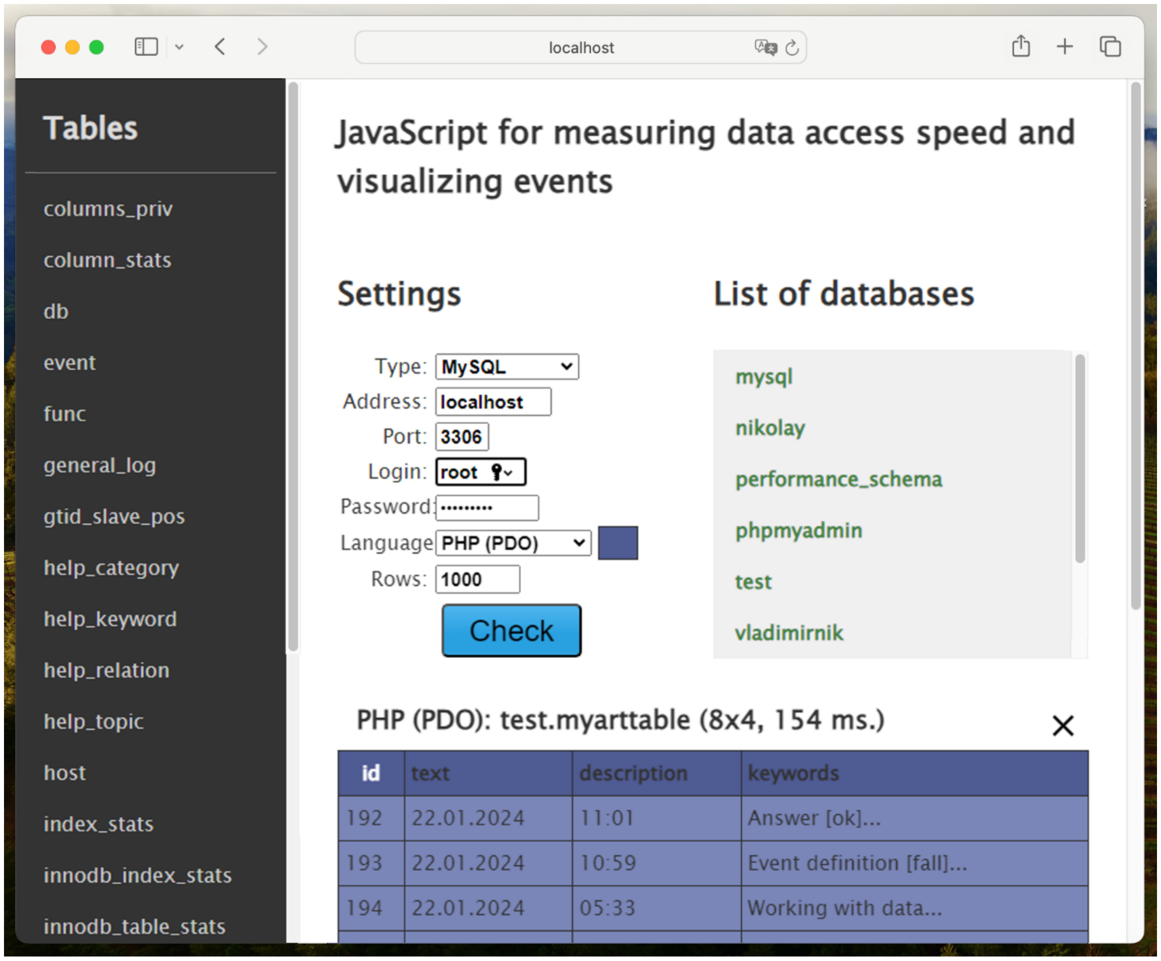


Figure 3: The main screen of the developed application for event visualization.

The authors of the research have also developed a website (FIGURE 4) that measures the time of data access and visualization of events in the intelligent video surveillance system using various DBMS and 18 different programming languages. The settings for connecting to the DBMS and the choice of a general-purpose script language are presented in the center of the website. On the right side, there is a list of databases on a single server instance, which is selected in the first drop-down list that is located in the center under the heading ”Settings”. On the left side there is a list of tables from the selected database. At the bottom of the site there is information about the used programming language and the used module for accessing the DBMS, the name of the table and the access time in milliseconds. The table ”<table> ... </table>” was created in the DOM via JavaScript, and the information was obtained via ”<iframe> ... </iframe>” without using jQuery and AJAX, only using HTML5 and JavaScript on the client side (Safari browser). CSS 3.2 styling was applied to the table in the form of classes that were connected via the element.classList.add(”my-class”) method from a single ”style.css” file and a script ”edit.js”. Bootstrap, Vue.js, Vanilla JS, Symfony, Angular, Flutter, Django, Laravel, Flask were not required for this website.

The TABLE 1, provides results of comparing of 17 scripting programming languages (PHP, Perl, Python, Go, Dart, JS, Ruby, Elixir, Lua, Swift, Java, Rust, D, R, Scala, Kotlin, Clojure) with SubScript when solving the tasks of storage and visualization events in intelligent video surveillance system. Three of the most popular operating systems were selected as the host system: macOS, Windows, and Linux. The latest distributions of scripting programming languages have been installed of these operating systems. Two operations were tested: write-read data and visualization of events on the site. Access time was measured using JavaScript on the site and accessing scripts in different languages via an ”iframe”. The time start was recorded at the moment of receiving the ”head” tag, and the time tracking was stopped when receiving the ”</html>” tag. The time was measured in milliseconds. During the research of programming languages used for solving two particular tasks, it was found out that the same logic (visualization and storage algorithm) is executed with different



Type:	MySQL
Address:	localhost
Port:	3306
Login:	root
Password:	•••••••••
Language	PHP (PDO)
Rows:	1000

PHP (PDO): test.myarttable (8x4, 154 ms.)

id	text	description	keywords
192	22.01.2024	11:01	Answer [ok]...
193	22.01.2024	10:59	Event definition [fall]...
194	22.01.2024	05:33	Working with data...

Figure 4: The developed website for measuring the speed of data access and visualization of events

Language name	MacOS	Windows	Linux	More than SubScript (% and at times)
PHP	73	71	74	**+19,34%** (0,36)
Perl	112	109	118	+59,67% (1,12)
Python	134	129	137	+80% (1,5)
Go (Golang)	98	93	102	+44,34% (0,83)
Dart	107	104	110	+53,67% (1,01)
Node.js	132	124	139	+78,34% (1,47)
Ruby	398	372	401	+337% (**6,32**)
Elixir	102	94	106	+47,34% (0,89)
Lua	231	215	247	+177,67% (3,33)
Swift	308	287	312	+249% (4,67)
Java	87	79	91	+32,34% (0,61)
Rust	94	89	102	+41,67% (0,78)
D (D lang)	219	198	225	+160,67% (3,01)
R (R lang)	127	109	137	+71% (1,33)
Scala	98	91	104	+44,34% (0,83)
Kotlin	93	89	97	+39,67% (0,74)
Clojure	109	102	113	+54,67% (1,03)
SubScript	54	49	57	0

Table 1: Comparison of code execution time (ms) of different languages for two tasks.

speeds. The SubScript language turned out to be faster than PHP by 19.34%, and the Ruby language by 6.32 times.

To reproduce the experiments, you need to install the programs according to the instructions [26], open the program logs in Microsoft Excel 365 for Mac, apply filtering, aggregation, average value and fill in the resulting table. A laptop with the following technical characteristics was used for measurements: 18" Laptop MSI Titan 18 HX A14VIG-096RU black 4K (3840 x 2400), mini-LED, Intel Core i9-14900HX, cores: 8 + 16 x 2.2 GHz + 1.6 GHz, RAM 32 GB, SSD 3000 GB, GeForce RTX 4090 for laptops 16 GB.

6. CONCLUSION

In the course of the research, the authors achieved an increase in the speed of data storage for the intelligent video surveillance systems using a new macro-based language. The new script language contains minimum of settings, does not consider configuration files, does not download additional libraries, and is very fast and easy to use. The language is intended only for solving the specific task.

Efficiency measurements were carried out relative to other programming languages, and allowed to conclude that the proposed solution for visualization and data storage for intelligent video surveillance systems that is proposed by the authors works faster by 19.34% compared to PHP, and 6.32 times faster than Ruby.

The solution was applied only to the intelligent surveillance system, however, it can be used in other systems where it is required to increase the speed of writing and reading from a database, to visualize events through an HTML5 template system.

References

[1] https://docsvision.com/.

[2] https://www.sap.com/.

[3] https://1c.ru/.

[4] https://sicon.ru/.

[5] Ierusalimschy R, Henrique De Figueiredo L, Filho WC. Lua—an Extensible Extension Language Software. Pract Experience;26:635-652.

[6] Cerqueira R, Cassino C, Ierusalimschy R. Dynamic Component Gluing Across Different Component Ware Systems. In: Proceedings of the International Symposium on Distributed Objects and Applications IEEE pp. New York: IEEE; 1999:362-371.

[7] Cowan DD, Stepien TM, Ierusalimschy R, Lucena CJ. Application Integration: Constructing Composite Applications From Interactive Components. Software. Pract Experience. 1993;23:255-275.

[8] https://rss2fido.sf.net/.

[9] http://old-dos.ru/.

[10] Sant'Anna F, Rodriguez N, Ierusalimschy R, Landsiedel O, Tsigas P. Safe System-Level Concurrency on Resource-Constrained Nodes. In: Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systemspp.ACM. 2013:1-14.

[11] Mascarenhas F, Medeiros S, Ierusalimschy R. On the Relation Between Context-Free Grammars and Parsing Expression Grammars. Sci Comput Program. 2014;89:235-250.

[12] Sudharsan S. Deep Learning-Based Intelligent Video Surveillance System for Real-Time Motion Detection. Int Sci J Eng Manag. March 2024;03:1-9.

[13] He F. Fangcheng .Intelligent Video Surveillance Technology in Intelligent Transportation. J Adv Transp. November 2020;2020:1-10.

[14] Li Y. A Survey on Edge Intelligent Video Surveillance With Deep Reinforcement Learning. J Intell. February 2022;7:70-83.

[15] Li J, Zheng Z, Li Y, Ma R, Xia ST. Multitask Deep Learning for Edge Intelligence Video Surveillance System. In2020 IEEE 18th International Conference on Industrial Informatics.IEEE. 2020;1:579-584.

[16] https://www.php.net/.

[17] https://www.apachefriends.org/.

[18] https://pecl.php.net/package/ssh2.

[19] https://www.python.org/.

[20] https://www.perl.org/.

[21] https://www.ruby-lang.org/.

[22] Tianxing M, Vodyaho A, Zhukova N, Subbotin A, Shichkina Y. Urban Intelligent Assistant on the Example of the Escalator Passenger Safety Management at the Subway Stations. Sci Rep. 2023;13:15914.

[23] Osipov V, Zhukova N, Subbotin A, Glebovskiy P, Evnevich E. Intelligent Escalator Passenger Safety Management. Sci Rep. 2022;12:5506.

[24] Man T, Osipov VY, Zhukova N, Subbotin A, Ignatov DI. Neural Networks for Intelligent Multilevel Control of Artificial and Natural Objects Based on Data Fusion: A Survey. Inf Fusion. 2024;110:102427.

[25] https://www.mongodb.com/.

[26] https://github.com/alex1543.