Technical University of Varna
Annual Journal
https://doi.org/10.29114/ajtuv.vol8.iss1.266
Vol. 8 Issue 1 (2024)
Published: 2024-06-28
ISSN 2603-316X (Online)

Development of programmable logic controller algorithms for database communication

Stefan Stefanov[1], *Anton* Naumov[2]

1 – Technical University of Varna, Department of Production Automation, 9010, 1 Studentska Street, Varna, Bulgaria
2 – Technical University of Varna, Department of Production Automation, 9010, 1 Studentska Street, Varna, Bulgaria

Corresponding author contact: slstefanov@abv.bg

Abstract. *The present paper provides a brief overview of the possibilities which can arise when outsourcing some of the more advanced functionalities usually developed for computers to a programmable logic controller. This approach is still being examined to fulfil its aim of improved communication capabilities and provision of effective tools for advanced intelligent control systems. The paper, further, sets out to give an example of one such algorithm which is built upon the MySQL protocol.*

Keywords: algorithm, communication, protocol, controller, MySQL

1 Introduction

Having access to much greater resources in the contemporary state of automation leads to developing a large number of control systems constrained by the higher computing power concentrated in the devices which indirectly take part in the control of the industrial processes - computers.

Traditionally, computers are considered a part of SCADA supervision systems due to the possibility to present large amounts of data in ways that are easy for the servicing staff to understand. The presence of such equipment on the spot also offers access to functionalities typically related to it like database operations, connection to the outside world through the means of the Internet, etc. The ability to extend the scope of the system and simplify the operation (which directly leads to a much shorter staff training time) is a perspective which few investors would fritter away considering the small financial aggravation of the project.

The main advantage of such systems is their ability to communicate with practically all the devices responsible for the proper control and monitoring of the complex industrial processes. This, of course, is not such an elementary task but it is achievable through the interpretation of the different communication protocols which each intelligent aperture uses. Some protocols are typically regarded as one-sided as they are traditionally used by one side and then translated and transformed in a suitable way for the next partner. Such are the database access protocols placing too great a load on the computers as those are initially the storage for the database information.

When using such structures, logic would lead one to the conclusion that the highest place in the hierarchy would be assigned to the computer followed by the local logical device which serves to control and monitor the lowest in the hierarchy, namely actuators and sensors. This kind of control is not without its drawbacks, one of which is the task of timely information supply. The computer is in a way the first requirement for the system's proper operation since it is the source of the information necessary for bringing the industrial process under control. Failure to communicate promptly the need for new data will inevitably result in the disruption of the working process which would not be able to continue its proper course.

2 Reversing the hierarchy

One possible solution to this task would be the reversal of the hierarchy which is proved possible given the advanced state of programmable logic controllers. This means that as far as database communication is considered, it is possible to move the queries directly to the controller and skip the translation and transformation of data from the computer as long as the controller has the tools to decipher

Technical University of Varna
Annual Journal
https://doi.org/10.29114/ajtuv.vol8.iss1.266
Vol. 8 Issue 1 (2024)
Published: 2024-06-28
ISSN 2603-316X (Online)

the database query answers. This task is not a particularly easy one to solve but overcoming it becomes achievable through the use of open-source software granting the developers the rights to use the necessary information in the future.

An example of such software is MySQL. In its essence, MySQL is a multithreaded multiuser SQL system for database management, (Viswani, 2004). Similar to other SQL systems, MySQL is used for operations with relational databases which in the last few years have proven to be more reliable and easier to operate due to the large coherence of the stored data. In the wake of its successful presence in the market, new developments were prompted in dozens of languages, thus, enabling MySQL to become accessible for a large number of computer applications, (Pajankar, 2020). This inclines to the establishment of a system in which the computer would be placed high in the hierarchy and would communicate the needed information to a controller through an open protocol, but still such a structure would not serve as a solution to the above-stated task of timely data supply. An additional shortcoming is the dependency of the actuators on the database connection with an additional communication unit which might pose a problem and lead to the disruption of the entire work process.

Using a controller to access a MySQL database, on the other side, is a problematic task due to the lack of freely distributed software to direct the implementation of the connection algorithm. It, then follows from here that if an access is to be provided, the entire algorithm with its predetermined set of instructions for connecting, sending, receiving, and decoding information from the database would be built from scratch in line with the MySQL protocol requirements. Such a task, undoubtedly, would take a lot of time and effort, and, therefore, it is first suggested to check whether an alternative approach is available, so that time, the most precious resource in the project, is not to be wasted away (Николов, 2013).

Building a system with a structure in which the positions are exchanged in this way would have to be reasoned and strongly encouraged. Indeed, drawbacks are already noticeable in the concept – such an idea demands resources which might not be available. A huge advantage, though, would be the possibility to use the results attained to further the advancement and re-use of the developed structure multiple times in other projects for as long as it is applicable. Since this, in its essence, is a gain for the contractor but neutral or even a loss for the investor, the emphasis should be shifted towards the other positive qualities of such a system. A good example is the independence from the computer as a control element, even though, in some respect, that role is retained – largely due to the greater availability of computing resources it is still the superior option for developing SCADA systems which in modern times are almost inseparable from industry management. Skipping over one of the communication units in the circuit will inevitably imply that the whole system will reach a higher work speed. Finally, such a structure leads to the solution of the foregoing task of timely data supply, as it would be unnecessary to communicate that need with the devices standing higher in the hierarchy.

3 Developing the algorithm

The requirements for implementing such a system are relatively reasonable – a programmable logic device, working under an open communication protocol, is required to send messages to the database server. The queries in question would have to meet a number of rules, set in place by the communication protocol, so they can be approved. This creates a demand for the capability to program the complex rules, under which the operations are validated, as those would at a later stage be used to decode the computer-received response. Required, in addition, are improved levels of performance so that the algorithms for preliminary query preparation and decoding can run in real time and without noticeable delay.

The requirements stated above do not pose a significant challenge for the larger part of modern technology. Many of the products available in the market especially those by some of the well-established brands for programmable logic controllers already meet these requirements and can be used for executing the task at hand. However, in order for the data to be visually presented through the programming environment in a way that resembles the database and allows the programmer to navigate intuitively through the messages received, the controller would have to be able to shape the information in a tabular view. That functionality is available by default in the Siemens controllers, (Berger, 2014).

Technical University of Varna
Annual Journal
https://doi.org/10.29114/ajtuv.vol8.iss1.266
Vol. 8 Issue 1 (2024)
Published: 2024-06-28
ISSN 2603-316X (Online)

This leads to the next question – how would one implement such a complex algorithm with the tools available in the programmable logic controller programming software? The first step is selecting the right programming language, as each one comes with its advantages and disadvantages, which should be considered prior to its appropriate development.

3.1 Ladder

Ladder, more commonly abbreviated as LAD, is the most widespread and easy-to-understand programming language in this field. It is a visual representation of the outdated relay-contactor techniques and can be used to simulate them, which leads to its main advantage – being easy to read and track. These qualities define it as the most suitable language for people with an extensive background in electrical engineering and the best language for detecting problems during runtime. This comes in pair with some drawbacks – as a result of the small amount of data contained in individual elements, complex expressions can take up large space. Due to the very nature of the language itself, this means that in these cases the networks, which represent the expressions will grow unidirectionally and hinder the complete portrayal of the created structure. This characterizes LAD as a rather lengthy language which is rarely used for developing complex algorithms, as the large volume of networks tend to make the navigation through the program rather complicated.

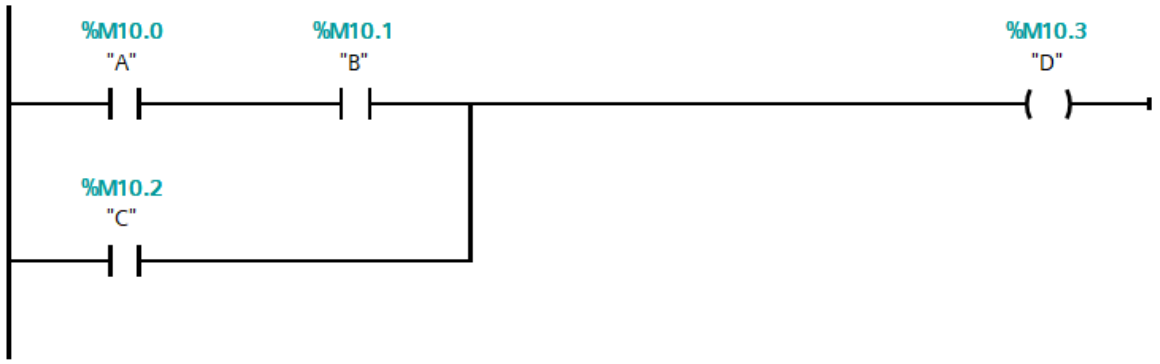

Fig. 1. Ladder network representing the function D = (A AND B) OR C.

3.2 Sequential function chart

A sequential function chart, more commonly abbreviated as SFC, represents a flowchart that uses the steps and transitions to bring about the desired results. The steps, similarly to the graphs, are the major functions and contain the actions that must be performed upon reaching them. Moving between steps demands a transition instruction which can be programmed to require some conditions to be met. Unlike traditional flowcharts, SFC can have multiple paths which is both an advantage and a disadvantage, as such functionality is its major selling point but at the same time does not always find use in traditional programmable logic controller applications. However, the language does allow the breaking down of the processes into smaller major steps which, in turn, makes troubleshooting much easier.

Technical University of Varna
Annual Journal
https://doi.org/10.29114/ajtuv.vol8.iss1.266
Vol. 8 Issue 1 (2024)
Published: 2024-06-28
ISSN 2603-316X (Online)

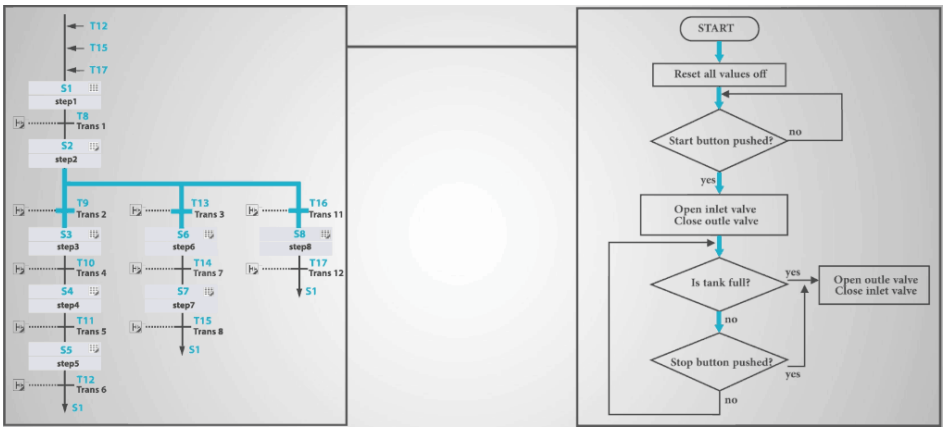

Fig. 2. A comparison between a sequential function chart(left) and a flowchart(right) (Retrieved from https://realpars.com/plc-programming-languages/)

3.3 Function block diagram

Function block diagram, more commonly abbreviated as FBD, is one of the most popular programming languages for programmable logic controllers, usually interchangeable with LAD. It is defined by a significantly higher density compared to LAD in view of the availability of ready-made functional blocks – elements which serve to complete a single specific task which is normally either clearly described in the programming guide or manually provided by the manufacturer. This withdrawal of information to an external source seems to indicate a more reduced need for a description inside the diagram itself, which is to explain why the FBD-based programs typically fit a lot more logical operations in a visually smaller part of the workspace. Alternatively, this allows for a much greater part of the program to be displayed. By using standard one-function blocks, it is easy to handle common, recurring tasks such as counters, timers, loops, etc, (Узунов, 2014). Tracking the program flow, however, is not always an easy task with the likely predisposition of the connections between the blocks to grow to incredible amounts and hinder the readability of the program even though it is possible to split those connections in two. It is precisely for this reason that the FBD is more often preferred by people with a high-level programming background who are well-accustomed to using ready-made functions and following separate threads running concurrently.

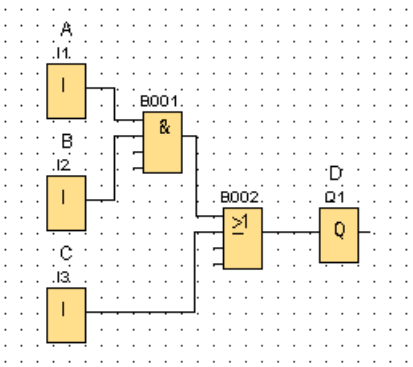

Fig. 3. Function block diagram representing the function D = (A AND B) OR C.

3.4 Structured text

Structured text, commonly abbreviated as ST, is a collective name for text-based programming languages for programming logical controllers. A common feature in these languages is that their use has high similarities with high-level programming although it is of the utmost importance during development to remember the difference between the processing power of computers and that of the controllers, so that resources are not exceeded. ST languages are usually based on PASCAL and allow access to all functionalities easily available in LAD and FBD in combination with others, ideally suit-

Technical University of Varna
Annual Journal
https://doi.org/10.29114/ajtuv.vol8.iss1.266
Vol. 8 Issue 1 (2024)
Published: 2024-06-28
ISSN 2603-316X (Online)

ed only in ST. This rich palette of possibilities while leading to smaller program sizes does not come without its drawbacks – finding a problem in the written code is hardly an easy task and spotting a logical mistake is time-consuming when compared to LAD. Accordingly, because of this, ST languages are more commonly used as a tool for developing function blocks and much less often as the main programming language.

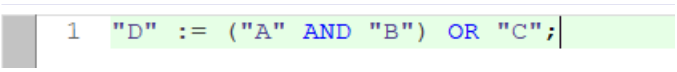

Fig. 4. Structured text representing the function D = (A AND B) OR C.

Despite the advantages and disadvantages of the languages under discussion, each one is applicable for developing a complex communication protocol but it is the ST that is considered the most suitable one due to its similarity with high-level programming languages which brings it closer to the software applications that have already been developed. However, in high-level programming languages, there are different techniques which can also have a great effect on the outcome of the created algorithm. In order to choose the right program structure, it is necessary to define which particular element of the expected algorithm can cause the most harm to its proper execution. Typically, when dealing with communication, the most serious issue is its failure which can happen due to either partner experiencing a performance problem or a long period of no data transmission leading to a disconnect. One way of circumventing this issue is by connecting and disconnecting the devices before and after each operation, which is a technique that has proven effective over the years when working with single-function equipment but in recent years due to the large number of obsolete operations is used occasionally and within certain limits.

Another issue which should be taken into account is that when working with a MySQL database while queries can be relatively short, the length of the answers is directly dependent on both the database and the query design which can produce an answer with millions of symbols in length. If the connection breaks down for one reason or another, this will lead to a serious issue, as part of the information may be missing and in order to resume operation, the exact moment of disconnection must be identified in order for all the appropriate measures to be taken. It is this stopping at key points and making transitions only under specific circumstances that is part of the *Finite-state machine* computation model being defined by its high stability based on the limited number of executions in a single key point, (Wang, 2019). Additionally, it is also easy to analyse any newfound problems since through this model it is always possible to note the step of the algorithm that was accessed last – a detail which is usually perceived as a weakness for ST programming but here such a benefit remains unmatched.

An example of the key features of the *Finite-state machine* model can be given in the Structured Text language:

```
IF (#Steps = #STEP_GREETING) THEN          //step entry point
   #Receive_greeting := TRUE;
   IF (#Receive_greeting.RCVD_LEN = 0) THEN
      #Receive_greeting := FALSE;
   ELSE
      #Steps := #STEP_LOGIN;
   END_IF;
END_IF;

IF (#Steps = #STEP_LOGIN) THEN             //step entry point
   //Form login request message
   …
   #Steps := #STEP_CHALLENGE;
END_IF;
```

While the example above is an incomplete part of an already developed and commissioned algorithm, it serves to show just how the *Finite-state machine* operates (in this case an acceptor subtype) – the *Steps* variable is used as a pointer to where exactly the program flow should go (Keller, 2001). In the code above *STEP_GREETING*, *STEP_LOGIN* and *STEP_CHALLENGE* are constant quantities to

Technical University of Varna
Annual Journal
https://doi.org/10.29114/ajtuv.vol8.iss1.266
Vol. 8 Issue 1 (2024)
Published: 2024-06-28
ISSN 2603-316X (Online)

ease readability and direct the algorithm to the right step; no other step can be executed with the entry points specifically checking the *Steps* variable ensuring that the program flows correctly.

4 Algorithm description

Having made the decision on how to implement a new communication protocol, securing the absolute minimum required for the successful transmission of data between two devices takes priority. It can be reasonably concluded, therefore, that a connection between the two communicating sides should be established first to be followed by a two-way data transmission, and at the end of the work process, an intentional disconnection has to occur so that the resources can be freed up and used to facilitate the work with a different partner. This can be generalized within the functions *Connect*, *Send*, *Receive* and *Disconnect*, (Abbate, 2000). Each of these functions is to be carefully described according to the requirements of the chosen protocol. In the case of the MySQL protocol, hereto discussed, the data transmission functions do not present a real challenge due to the comprehensive description received along with each block of data that is extracted from the database. As for the *Disconnect* operation, it is made up of only a standard FIN packet according to the TCP/IP protocol requirements, which is acknowledged by the partner side with an ACK segment. This is practically a repeat of the two-way data transmission functions but much simpler and usually, part of ready-made functions within the programming environments.

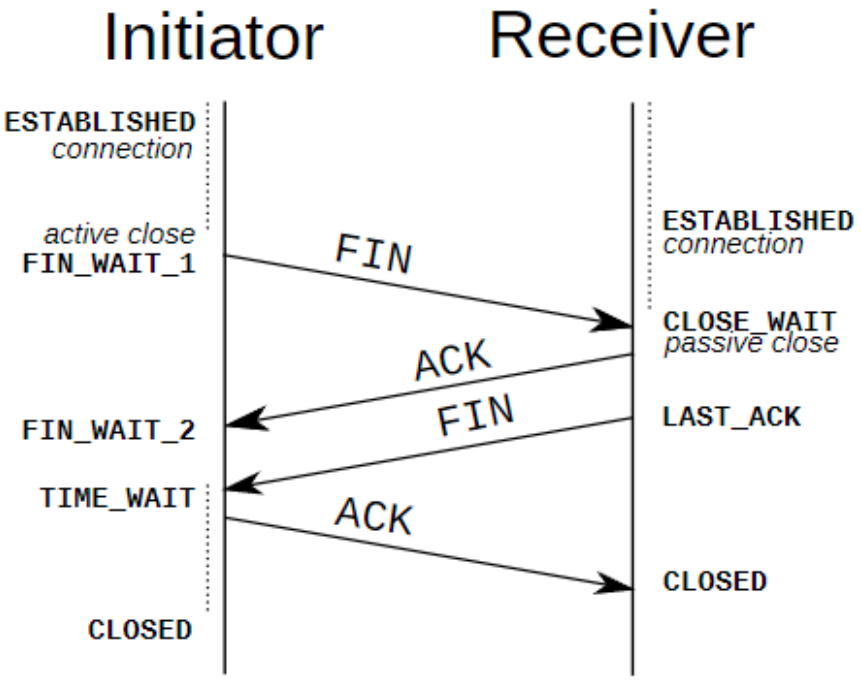

Fig. 5. TCP connection termination procedure.

This brings attention to the *Connect* function which for the MySQL protocol is the most complex one of the four. This is a consequence of the fact that this type of communication grants access to a large amount of data which may even be confidential. Because of this, connection privacy is of the highest priority, and it is mandatory to guarantee that the device, trying to access the database, truly has the full information needed to be granted entry and not just partial fragments which may suggest that this is, in fact, a malicious attack.

For the protocol under study, the procedure must be followed as:

- Starting a communication according to the TCP protocol with the MySQL server
- Awaiting handshake signal
- Forming a request to be granted entry according to the protocol's requirements
- Awaiting an access key

Technical University of Varna
Annual Journal
https://doi.org/10.29114/ajtuv.vol8.iss1.266
Vol. 8 Issue 1 (2024)
Published: 2024-06-28
ISSN 2603-316X (Online)

- Preparation of an answer containing the password for the chosen user and the access key in a secure form
- Awaiting authenticity confirmation

The structure of the algorithm is an upgrade of the well-developed TCP communication and expands it with the only purpose of guaranteeing security and access control. The first step, starting a communication, is a feature of the well-known handshaking that is part of the TCP protocol and subject to the MySQL protocol version it can be either a three-way or a four-way handshake. The obvious difference is the number of messages that have to be transmitted between the two sides. Initiating communication through a TCP protocol is done by two-way requests using a SYN message followed by a two-way agreement using an ACK message. In total, that is 4 messages that both sides should exchange before it is confirmed that communication is, in fact, started and data transmission can begin. Since one side will always act as a client and the other one as a server, one of the SYN messages will inevitably arrive at a seemingly random moment. This allows the existence of the three-way handshake, which means that after receiving the client's request for starting a connection, the server can send a SYN-ACK message to save the need for a fourth message in the communication startup. It is important to note that a standard part of this procedure is communicating the sequence number tied to the connection and it must be incremented to ensure that there will not be a conflict in identifying partners on either side, (Cerf, 1983).

Part of the upgrade, done by the MySQL protocol, consists of the presence of an automated greeting done by the server once the handshake ends. This greeting is part of a separate handshaking protocol which is developed by MySQL AB and isn't used in the same iteration in other similar products. The point of this improved handshake is to transmit the details tied to establishing successful communication with the database such as the version of the protocol in use, connection identification number, character set, etc. Before this greeting is transferred, the handshake protocol version is announced so that the client may know which message decoding protocol has to be used.

After the client has received the necessary information to access the database, an entry request has to be formed and issued according to the requirements of the protocol in use. Semantically, this request serves as a confirmation that the client indeed has the tools to correctly decipher the server-received messages but also provides the server with the necessary information for client recognition. Due to the nature of the introduction, it is possible to fabricate this information although providing false data can only lead to problems in the future therefore it is highly suggested the client introduction is done properly in a sensible and comprehensible manner. One of the mandatory fields is the name of the user by which the client wishes to be identified. If a given user does not exist in the database, communication is terminated as this serves as a proof that this particular client does not have the complete data required to be granted the right of entry.

Upon recognizing the username and confirming the login request, the server initiates a password request. In modern times, computers have large enough computing resources to be able to launch attacks against different structures and in doing so, sidestep the password request. It is for this reason that in recent years a growing number of safety measures are being introduced to ensure that upon a simple observation of the exchanged messages, delicate information like passwords is not presented in plain text. While it seems obvious, even today thousands of public sites break this unofficial rule and are therefore not considered trustworthy. The MySQL protocol attempts to propose a solution to this problem by means of two steps – provision of an access key which serves as a challenge to generate the correct answer in a limited amount of time along with the additional hash functions for further protection.

By definition, hash functions are one-way mathematical functions which serve to associate data to its hash value which is used as an index key of fixed length, (Knuth, 1973). Upon passing the exact same input, the hash result would always be the same which leads to the conclusion that if two inputs produce the same hash values then they ought to be identical. Theoretically, it is possible for the two different blocks of data to produce the same hash values. This is called a collision and is a leading topic in computer science, (Stapko, 2007). Generating a collision intentionally depends directly on the length of the generated hash function output through which recent functions with a higher level of security like SHA256 and SHA512 are being used. However, generating a collision attack requires time and even when large computing resources are available, a security breach may not always be pos-

Technical University of Varna
Annual Journal
https://doi.org/10.29114/ajtuv.vol8.iss1.266
Vol. 8 Issue 1 (2024)
Published: 2024-06-28
ISSN 2603-316X (Online)

sible if unique data is present, (Soltanian, 2016). Because of this, the MySQL protocol and many others rely on the usage of a temporary access key which is part of the input data for the hash function. By means of a constantly changing input and a time control for generating the answer, the possibility for a malicious attack is minimized – if a correct answer is not received in time or an incorrect one is sent instead, it is determined that the client does not, in fact, have the information needed to be granted access to the database and the communication is terminated immediately.

For the purposes of the MySQL protocol, a SHA1 hash function is being used. It is characterized by an output length of 20 bytes which are broken into 5 words of 4 bytes (32 bits). It is mandatory that the input is a multiple of 512 bits which can be broken down into 16 32-bit big-endian words which would be of the same length as those mentioned above. This means that in the case of larger messages hashing would be done in separate 512-bit chunks. The hash function itself consists of 80 looping steps, each 20 of which follow a different algorithm and have different constants. This leads to a complete transformation of the original message and the obtained result cannot be traced back to the input unless the function is executed again. The function is irreversible which means that even though messages are coded in a much smaller volume relative to the modern SHA256 and SHA512, it is still necessary for all the initial conditions to be known in order for the same result to obtained.

The authorization algorithm consists of multiple hash functions to circumvent the security weakness which SHA1 poses (Stevens et al., 2017) and by definition is:

$$SHA1(password)\ XOR\ SHA1(challenge\ \langle concat \rangle\ SHA1(SHA1(password))) \quad (1)$$

Formula (1) shows that it is compulsory for both the password and the server-given access key (challenge) to be known. Some may notice that the two instances of the SHA1 hash function have different input data lengths, but this is not an issue as the hash output is of fixed length which means the logical operation XOR will always be performed bit by bit without any problems. The only challenge would be to ensure effective local implementation of the SHA1 algorithm in the logic device responsible for the communication itself. If the result obtained through formula (1) matches the result calculated by the server, then access is granted and database operation may begin.

5 Conclusion

Delegating more responsibilities to the device with lower computational power can in some cases have positive effects on the project as a whole instead of being a net negative. Removing the extra communication unit from the network leads to an increase in performance and stability, ensuring independence between devices, and authorizing them to execute operations under fewer requirements. As a whole, the task of developing algorithms for managing the process that was originally placed in the now obsolete device is a laborious one and rarely can be justified for single applications. As a long-term investment, however, and further development of the algorithm as a ready-made functionality in other projects is a common practice when undertaking an intelligent approach towards creating control systems and managing the most limited resource of all – time.

Acknowledgments

The research was conducted under the project PD4/21 named “Developing and researching algorithms for the extraction, processing and protection of data” within the academic activities inherent to TU-Varna and with the financial support from the state budget for the 2021 research project program.

References

Wang, J., & Tepfenhart, W. (2019). *Formal Methods in Computer Science*. CRC Press. https://doi.org/10.1201/9780429184185

Keller, R. (2001). *Classifiers, Acceptors, Transducers, and Sequencers*. Retrieved from https://www.cs.hmc.edu/~keller/cs60book/12%20Finite-State%20Machines.pdf

https://doi.org/10.29114/ajtuv.vol8.iss1.266
Vol. 8 Issue 1 (2024)
Published: 2024-06-28
ISSN 2603-316X (Online)

Vaswani, V. (2004). *MySQL™: The Complete Reference*. McGraw Hill India.

Pajankar, A. (2020). *Learn SQL with MySQL*. BPB Publications. Retrieved from https://www.perlego.com/book/1681500/learn-sql-with-mysql-pdf (Original work published 2020)

Cerf, V. G., & Cain, E. (1983). The DoD internet architecture model. Computer Networks (1976), 7(5), 307-318. https://doi.org/10.1016/0376-5075(83)90042-9

Abbate, J. (2000). *Inventing the Internet*. MIT Press.

Димитров, В., Николов, Н., & Александрова, М. (2013). *Автоматизация на технологични процеси*. ТУ Варна.

Петров, П., & Узунов, В. (2014). *Архитектура на системите с PLC SIMATIC S7*. ТУ-Варна. ISBN 978-954-20-0659-6

Stapko, T. (2007). *Practical Embedded Security*. Elsevier. https://doi.org/10.1016/B978-075068215-2.50006-9

Soltanian, M., Amiri, I., & Neeley, M. (2016). *Theoretical and Experimental Methods for Defending Against DDoS Attacks*. Syngress.

Stevens, M., Bursztein, E., Karpman, P., Albertini, A., & Markov, Y. (2017). *The First Collision for Full SHA-1*. https://doi.org/10.1007/978-3-319-63688-7_19

Knuth, D. (1973). *The Art of Computer Programming, Vol. 3, Sorting and Searching*. Addison-Wesley.

Berger, H. (2014). *Automating with SIMATIC S7-1500*. Publicis MCD Werbeagentur GmbH.

Online sources

https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html

https://dev.mysql.com/doc/dev/mysql-server/latest/page_protocol_connection_phase_packets_protocol_handshake_v10.html