EXPERIMENTAL COMPARISON OF REST AND SOAP WEB SERVICES FOR REAL-TIME FACE RECOGNITION

Haia Kablan [1]

[1] Latakia University, Latakia, Syria, haiakabalan3@gmail.com

Lilia I. Voronova [2]

[2] Moscow Technical University of Communications and Informatics, Moscow, Russia

DOI: 10.36724/2664-066X-2026-12-2-26-35

Received: 10.02.2026
Accepted: 12.04.2026

Citation: Haia Kablan, L.I. Voronova, "Experimental comparison of rest and soap web services for real-time face recognition," *Synchroinfo Journal* **2026**, vol. 12, no. 2, pp. 26-35.

ABSTRACT

The primary objective of this study is to identify the optimal platform for IoT applications with limited resources and real-time requirements. With the development of the Internet of Things, interest in performance testing of web platforms supporting these applications has increased. This study compares REST and SOAP technologies for use in a smart home with facial recognition. Experiments were conducted on Raspberry Pi OpenCV, and testing was performed on JMeter. The results showed significant improvements in metrics: REST throughput was 1.5-5.2 times higher, latency was 2.0-2.5 times lower, and errors were reduced by 50% compared to SOAP. Based on the data obtained, an optimized web service architecture was proposed for an intelligent real-time monitoring system for a smart home environment.

KEYWORDS: *Internet of Things (IoT), face recognition, Raspberry Pi, OpenCV, Haar cascade algorithms, LBPH (Local Binary Patterns), Web services, REST SOAP.*

Introduction

With the development of the Internet of Things (IoT), interest in performance testing of web platforms supporting its applications has increased. This paper compares the performance of RESTful and SOAP APIs using a Raspberry Pi-based intelligent video surveillance system implemented using facial recognition technologies (Haar cascade and LBPH).

The primary goal of this study is to determine the optimal platform for resource-constrained IoT applications with real-time requirements.

Previous research in this area

Researchers identify three areas of research related to object recognition in smart home design. First, they address the problem of practical implementation of facial recognition systems for real-world smart home projects. The approach described in [1-4] involves using a Raspberry Pi microcomputer with computer vision algorithms such as Haar Cascade/LBPH. This approach can be considered the basis for developing functional prototypes.

The second area of research includes work on the architecture and communication of IoT platforms, with a particular focus on device integration, interaction paradigms, and security. The main conclusion of [5, 6] is the key role of a universal "link" between devices and the user.

The third area of research explores and compares two protocols used for developing web services (REST and SOAP) in the IoT. The works are theoretical or demonstrative in nature, discussing the advantages of the REST protocol (simplicity, efficiency) and presenting models of its application [7-9]. There are no practical tests of protocols in real-world IoT conditions, with limited resources and high protocol speed requirements.

As is well known, the operation of the Internet of Things involves a number of integrated stages, such as collecting environmental data, processing and decision-making, control actions, storage, and analysis [10].

The Department of Information Systems and Automation at MTUCI is developing a number of projects related to the application of intelligent technologies in the fields of the Internet of Things, computer vision, and embedded systems cybersecurity [11-13]. These projects include the development of intelligent monitoring systems, performance analysis of communication protocols under limited resources [14], and the development of automated methodologies for improving the reliability and efficiency of intelligent systems in real time. This research is carried out practically using prototypes and performance tests with the goal of creating innovative solutions [15].

Recognition technologies

The facial recognition process consists of three stages: face recognition using classifiers, feature extraction, and identification by comparison.

Face Recognition Using the Haar Cascade Classifier

A pre-trained classifier (haarcascade_frontalface_default.xml) is used for efficient real-time face recognition [16]. The method is based on Haar features, which are calculated as the difference between the sum of pixel intensities in adjacent rectangular regions (Fig. 1).

Fig. 1. Haar Feature [17]

Value = ∑(pixels in black region) - ∑(pixels in white region)……….…….   (1)

To speed up the computations, an integral image is used, as shown in (b), which allows the sum of pixels in any rectangle to be calculated as follows:

$$\mathbb{S}(D) = II(4) - (II(3) + II(2)) + II(1).....(b)[17] \quad (2)$$

where S(D) is the sum of the pixels in rectangle D alone and is the sum of the pixels in rectangles A + B + C + D, represented as II(4); II(3) is the integral image of rectangles A + C; II(2) is the integral image of B + A; and finally, II(1) is the integral image of rectangle A. The addition is performed due to the fact that the integral image is defined as follows:

$$II[x, y] = I[x', y'].....(c)[13] \quad (3)$$

where II[x,y] represents the original image, and I[x', y'] is the entire image.

The Adaboost algorithm is used to select the most significant features from ~160,000 possible features, reducing them to ~6000 [17-20]. A cascade approach is used to quickly remove non-edge regions [17-21].

Feature Extraction with LBPH

After face recognition, facial features are extracted using the LBPH (Local Binary Patterns Histograms) algorithm with the following parameters: radius = 1, neighbors = 8, 8x8 grid [21].

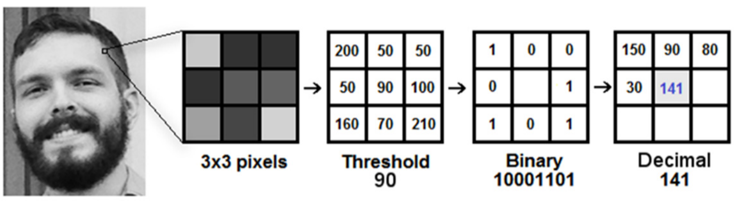

Fig. 2. LBP Algorithm [20]

For each pixel, a binary pattern is calculated (Figure 2), local histograms are calculated, and then combined into a descriptor with 16,384 dimensions (8x8x256).

Face Identification

Face identification is performed by matching the LBPH descriptor of a new image against the processed database, based on Euclidean distance. The output provides a face identifier and a confidence level [21].

Web Platform and Services

The developed platform is based on a three-tier architecture consisting of a server part, a user interface, and APIs for interaction between them, with support for both fundamental web service paradigms: REST and SOAP [22]. The service uses a standard architecture based on a service provider, consumer, and registry [23] and is characterized by loose coupling, flexibility, reusability, interoperability, and automatic discovery [24]. SOAP is a standardized XML-over-HTTP messaging protocol that provides platform independence and security, and its message structure consists of an envelope, header, body, and error element [25]. Meanwhile, REST is an architectural style for developing web applications that relies on the basic HTTP methods (GET, POST, PUT, DELETE) and returns data in JSON-like formats based on the principles of client-server separation, statelessness, caching, a uniform interface, and a multi-tier system [26-27].

Implementation of a Web Service Comparison Monitoring System Prototype

Using a three-tier architecture, a real-time web service monitoring system for a smart home environment was developed. The camera module and HC-SR501 motion sensor are installed on a Raspberry Pi 3 Model B+ device at the Edge tier (Fig. 3).

Fig. 3. Final connection setup

The device runs the Raspberry Pi Buster operating system and was configured to operate as an IP camera using motion tracking software, enabling streaming. H.264 video with a resolution of 640x480 pixels and a frame rate of 15 frames per second is transmitted via port 8081 on the local area network (LAN).

The main server, which collects HTTP streams from the Raspberry Pi, runs Ubuntu 20.04 LTS, a Core i7, and 12 GB of RAM. Django 3.2.4 and OpenCV 4.5.1 are used for video processing.

Each video frame is processed to perform face recognition using the Haar cascade method, feature extraction using the LBPH method with parameters radius=1, neighbors=8, grid=8x8, and comparison with a database of 500 images of 50 faces, saving the result in SQLite.

The web interface allows you to register, log in, and interact with the dashboard. The dashboard has two main actions.

1. Create a service: For the service creation action, the user can select either real-time face recognition or video face recognition.

2. Training data: For the training data action, the user can select Train the system and upload images and names for face recognition.

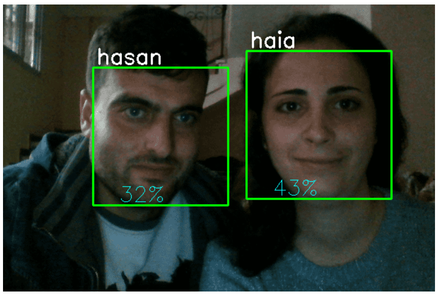

Fig. 4. Live Page View Result

The results are displayed on the live stream page (Fig. 4).

Users can view live video streams and perform facial recognition.

API/service/<str:API_key>
API/service/faces/<str:API_key>
API/face/<str:fid>

Fig. 5. Endpoints Provided by the Two Platforms

The architecture of the system using the two developed platforms, REST and SOAP, is shown in Fig. 5.

System Testing and Performance Evaluation

To test and measure the relative performance of the RESTful and SOAP APIs, Apache JMeter was used, simulating a series of user interactions in a predefined testing environment. Four parameters were considered when evaluating RESTful and SOAP API-based applications: throughput, which measures the ability of a single system to process multiple requests within one second; initial response time, latency; average overall response time; and error rate. Each parameter was implemented 30 times to obtain an average value.

The first scenario involves varying the number of users (10, 100, 200, 500, 1000) attempting to access the getfaces service, which returns information on 250 images with a ramp-up time of 10 seconds, and calculating the error rate for 5 request iterations.

A group of threads executes the same scenario. The number of threads is the number of virtual users that can connect to the server and was set to 50, 100, 150, and 200, respectively.

The ramp-up time is the amount of time it will take Apache JMeter to add all tested users during the test.

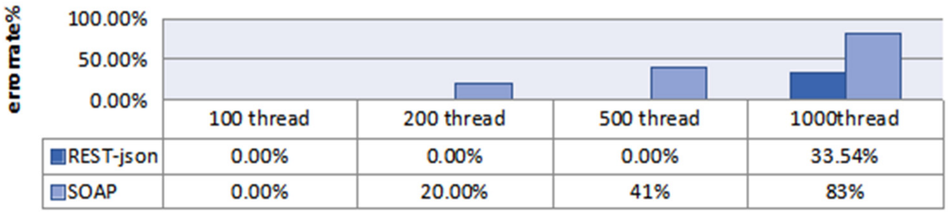

Fig. 6. Error rate for 250 photos using getfaces

The REST platform can handle 1,000 users with 5,000 requests (with 5 retries) and has an error rate of 33.54%, while the SOAP error rate is 83%, as shown in Figure 6.

The second scenario involves changing the number of images (100, 150, 200, 250) in the face collection for 100 users accessing the getfaces service and calculating the response time and throughput with 5 retries. The ramp-up time is 10 seconds, meaning 10 users are added every second. Each time, the number of images returned by getfaces changes, and the requests are retried 5 times, resulting in a total of 500 requests.

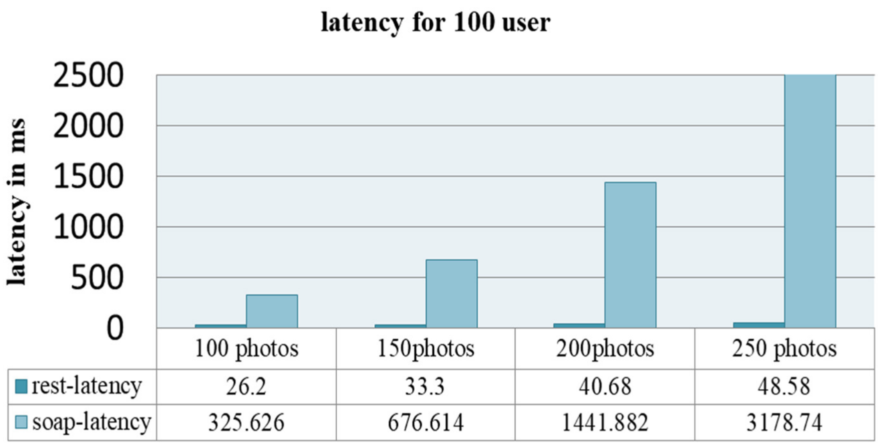

Figure 7. Latency in the Second Scenario

We note that as the response size increases from 100 to 250 images, there is a significant difference in latency between the SOAP-based platform and the REST-based platform, as shown in Figure 7.

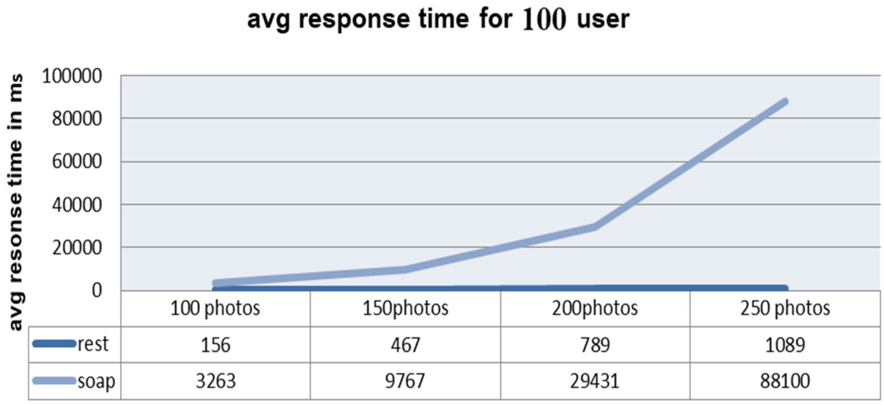

Figure 8. Average response time for the second scenario

As Figure 8 shows, REST requires less than 1 second for 250 images, while SOAP takes approximately 88.1 seconds, which is a significant difference.

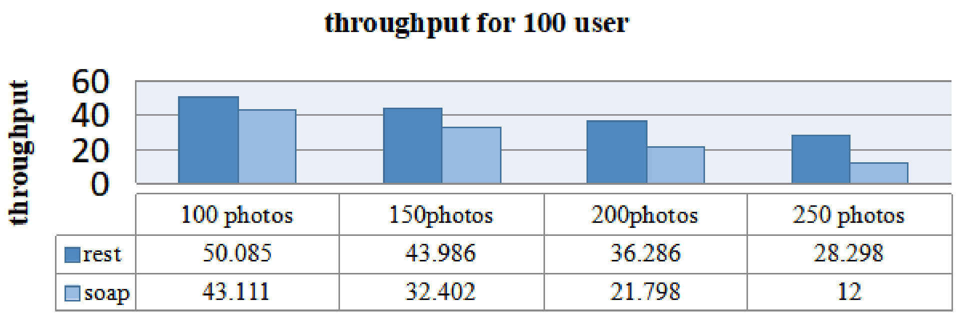

Figure 9. Performance for the Second Scenario

Figure 9 shows the performance for the second scenario. These results indicate that REST yields the best results.

Third Scenario: Launching 15 live video streaming services (15 sensors) and providing access to the service to 500 users with a startup time of 10 seconds.

Table 1

Results of the Third Scenario

Label	# Samples	Average	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes	latency
rest	500	20.9	50.44022	23.27	6.06	492	20.676
soap	500	27	49.5491	37.16	23.61	768	26.948

This scenario allows users to monitor more than one room. Table 1 illustrates the results of this scenario.

Scenario Four: Spike Testing

Spike testing analyzes the system's response to unexpected, sharp increases or decreases in system traffic.

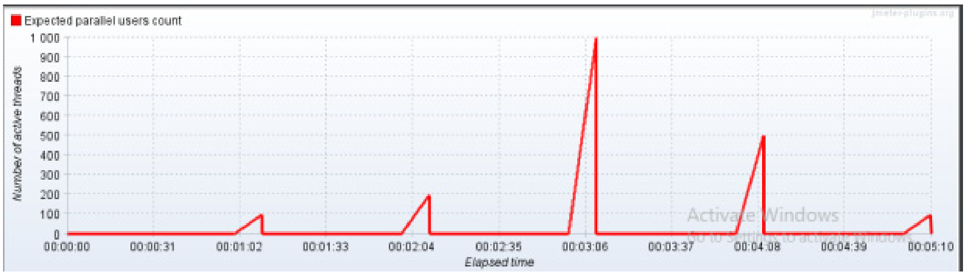

Figure 10. Configuration Jump Testing

Figure 10 below shows an example of flow testing using a finite set of flows in JMeter with various parameters: Initial Number of Users, Initial Latency, Startup Time, Load Hold, and Shutdown Time.

Table 2

Results of the Fourth Scenario

Label	# Samples	Average	Min	Max	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
soap	14771	953	6	6860	39.45%	47.57518	69.81	13.73	1502.5
Rest	21504	556	4	1077	43.14%	69.27768	95.69	4.92	1414.4

It is noted that the actual number of requests also depends on the response time. These test results are presented in Table 2.

Conclusion

An experimental study demonstrates the superiority of using a REST architecture over SOAP, achieving a 1.5x–5.2x throughput increase, a 2.0x–2.5x latency improvement, and a 50% improvement in error rate. This demonstrates that this efficiency is due to its ability to effectively manage limited resources and process intelligent IoT applications in a timely manner. It is recommended that this effective methodology be standardized for intelligent IoT applications. Furthermore, this system has the potential to effectively serve up to thousands of cameras in a commercial environment, leverage AI and deep learning techniques, and utilize various APIs to meet varying energy needs.

REFERENCES

[1] R. Kazi, G. Chaudhary, “Live Video Streaming using Raspberry PI with Face Detection,” *International Journal of Engineering Research & Technology (IJERT),* 2019. Vol. 8, No. 11. November, pp. 716-717.

[2] F. Rahamanm, A. Al Noman, M. Ali, M. Rahman, “Design and implementation of a face recognition—based door access security system using Raspberry Pi,” *International Journal of Research in Engineering and Technology (IRJET).* 2021. Vol. 8, No.11. December, pp. 1705-1709.

[3] D. Mali, R. Patil, N. Dharwadkar, Ch. Devale, O. Tembhurne, “Real-Time Smart Surveillance System Using Raspberry Pi,” *SSRN Electronic Journal.* 2019. January, pp. 1851-1857.

[4] R. Novosel, B. Meden, Z. Emer, V. Struc, P. Peer, “Embedded Engineering IoT – face recognition with Raspberry Pi,” *ResearchGate.* 2017. September 8.

[5] Z. Yang, X. Guo, D. Janowsky, X. Guo, C. Chang, “A web platform for globally interconnected 6Lowpan networks,” *Proceedings of the 2019 International Conference on Embedded Wireless Systems and Networks*. Beijing, China, 2019. 25-27 February, pp. 367-372.

[6] S. Cavalieri, G. Cantali, A. Susinna, “Integration of IoT technologies into smart grid systems,” *Sensors.* 2022. Vol. 22, No. 7. March. DOI: 10.3390/s22072475.

[7] R. Maurya, K. Nambiar, P. Babbe, J. Kalokhe, Y. Ingle, N. Shaikh, “Application of Restful APIs in IoT: A Review,” *International Journal for Research in Applied Science and Engineering Technology (IJRASET).* 2021. Vol. 9, no. 2, pp. 145 -151. DOI: 10.22214/ijraset2021.33013.

[8] L. Pan, M. Xu, Xi L., Y. Hao, “Research of Livestock Farming IoT System Based on RESTful Web Services,” *5th International Conference on Computer Science and Network Technology (ICCSNT).* 2016, pp. 45-50.

9. F. Halili, E. Ramadani, “Web Services: A Comparison of Soap and Rest Services,” *Modern Applied Science.* 2018. Vol. 12, no. 3. P. 175. DOI: 10.5539/mas.v12n3p175.

[10] Leverage. How IoT Systems Work [Electronic resource]. IoT eBook. URL: https://www.leverege.com/iot —ebook/how —iot —systems —work (date of access: 3.12.2025)

[11] O. L. Antonycheva, L. I. Voronova, “Teaching the discipline "Machine Learning" using visualization tools,” *Technologies of the Information Society: Coll. proc. XVIII Int. industry scientific and technical. conf.* (Moscow, February 27-28, 2024). Moscow: Moscow Technical University of Communications and Informatics, 2024, pp. 369-371.

[12] A. G. Vovik, L. I. Voronova, "Methodology of automated management of information security in the Internet of Things systems," *H&ES Research*. 2024. Vol. 16. No. 4, pp. 4-11. doi: 10.36724/2409-5419-2024-16-4-4-11.

[13] K. A. Kalushev, L. I. Voronova, “Implementation of an approach to determining the coordinates of objects in the field of technical vision,” *DSPA: issues of digital signal processing application*. 2024. Vol. 14, No. 3, pp. 30-36.

[14] N. F. Mohammad, L. I. Voronova, V. I. Voronov, S. A. Rozhkov, “Software complex for modeling routing in a heterogeneous model of wireless sensor network,” *Systems of signals generating and processing in the field of on-board communications*. 2024. Vol. 7, no. 1, pp. 281-285. ISSN 2768-0096. eISSN 2768-0118.

[15] V. A. Smolnikov, L. I. Voronova, V. I. Voronov, S. A. Rozhkov, V. M. Petukhov, “Simulation of the digital twin of the technological process of creating a demonstrator using R-PRO digital,” *Systems of signals generating and processing in the field of on board communications*. 2024. Vol. 7, No. 1, pp. 438-442. ISSN 2768-0096. eISSN 2768-0118.

[16] S. Singh, P. Anap, Y. Bhaigade, J.Chavan, “IP Camera Video Surveillance using Raspberry PI,” *Journal of Advanced Research in Computer and Communication Engineering (JARCCE).* 2015. Vol. 4, No. 2. February, pp. 326-328.

[17] M. Raoa, H. Palleb, P. Dasaric, Sh. Jannaikode, “Implementation of Low Cost IOT Based Intruder Detection System by Face Recognition using Machine Learning,” *Turkish Journal of Computer and Mathematics Education.* 2021. Vol. 12, no. 13, pp. 353-362.

[18] R. Senthamizh, D. Sivakumar, S. Sandhya, S. Siva, S. Ramya, K. Suba, S. Raja, “Face Recognition Using Haar-Cascade Classifier for Criminal Identification,” *International Journal of Recent Technology and Engineering (IJRTE).* 2019.Vol. 7, no. 6S5. April, pp. 1871-1876.

[19] Willberger. Deep learning: Haar-cascade explained [Electronic resource]. 2022. November 26. URL: www.will-berger.org/cascade — haar — explained (date of access: December 5, 2025).

[20] W. Zhao, R.Chellappa, “Face Recognition: A Literature Survey,” ACM Computing Surveys. 2003. Vol. 35, No. 4. December, pp. 399-458.

[21] P. Singh, “Understanding Face Recognition Using LBPH Algorithm [Electronic resource],” *Analytics Vidhya*. 10.21.2024. URL: https://www.analyticsvidhya.com/blog/2021/07/understanding-face-recognition-using-lbph-algorithm/ (access date: 12/27/2025).

[22] C. Reiff, S. Oechsle, F. Eger, A. Verl, “Web-based Platform for Data Analysis and Monitoring,” *Procedia CIR*P. 2019. Vol. 86. January, pp. 31-36.

[23] M. Gashti, “Investigating Soap and Xml Technologies in Web Service,” *International Journal on Soft Computing (IJSC).*

[24] M. Govindaraju, A. Slominski, K. Chiu, P. Liu, R. Engelen, M. Lewis, “Toward characterizing the performance of SOAP toolkits,” *Proceedings of the 5th IEEE/ACM International Workshop on Grid Computing*. Pittsburgh, PA, 2004. November, pp. 365-372.

[25] C. Kiama, L. Muchemi, “Comparative Study of REST and SOAP: Case of Registrar of Political Parties’ Kenya,” *Trends in Distributed Computing.* 2014. January, pp. 105-116.

[26] R. Sinha, M. Khatkar, S. Gupta, “Design & Development of a REST based Web Service Platform for Applications Integration on Cloud,” *International Journal of Innovative Science, Engineering & Technology (IJISET).* 2014. Vol. 1, Iss. 7, pp. 385-389.

[27] M. Agarajan, Ch.Raveendra, “Role of web service in the internet of things,” *Proceedings of the International Conference on Applied and Theoretical Computing and Communication Technology (IEEE)*. 2017. 21-23 December, pp. 21-23.