DOI:10.25092/baunfbed. 929483 *J. BAUN Inst. Sci. Technol., 24(1), 251-262, (2022)*

Improving the delay and service blocking performance of a global trade, import/export, transit and logistics management system & software based on microservices architecture

Necla BANDIRMALI ERTÜRK [1,*], Yasemin HOCAOĞLU [2]

[1] *Bandırma Onyedi Eylül Üniversitesi, Bilgisayar Mühendisliği Bölümü, 10145, Balıkesir, Türkiye*
[2] *Proaktif Dijital Yönetim ve Eğitim Sistemleri, Kocaeli Üniversitesi Teknopark, Kocaeli, Türkiye*

Geliş Tarihi (Received Date): 28.04.2021
Kabul Tarihi (Accepted Date): 04.11.2021

Abstract

A scalable and microservices architecture-based system & specific software for a complete Global Trade - Import/Export Management (GTM) solution is presented in this paper. It has been developed by means of a modular web application approach, containing functionally different tools such as customer management, business process management, accounting & finance management and marketing & sales management components that all can be adapted to any other business sector. It allows for any user/system-required expansion through a cloud-based and flexible modular structure, proposing a framework for commercial solutions. Compared to its classical monolithic-based counterpart, it well offers above 10 times better delay results and remarkably decreases the system blocking ratio.

Keywords: *Microservices, scalability, modular web application, management software, global trade management system*

[*]Necla BANDIRMALI ERTÜRK, nerturk@bandirma.edu.tr, http://orcid.org/0000-0002-3513-7846
Yasemin HOCAOĞLU, yaseminhocaoglu@proaktif.org, http://orcid.org/0000-0001-8036-3909

Mikroservisler mimarisi kullanılarak global ticaret, ithalat/ihracat, transit ve lojistik yönetim sistem & yazılımı gecikme ve hizmet tıkanma başarımının iyileştirilmesi

Öz

Bu makalede entegre Global Ticaret - İthalat/İhracat Yönetimi (GTY) çözümü için ölçeklenebilir ve mikroservisler mimarisi tabanlı bir sistem ve özel yazılım sunulmaktadır. GTY, müşteri yönetimi, iş süreçleri yönetimi, muhasebe-finans yönetimi ve pazarlama-satış yönetimi bileşenleri gibi işlevsel olarak farklı araçlar içeren modüler bir web uygulaması yaklaşımı ile geliştirilmiştir. Ticari çözümler için bir çerçeve de öneren bu çalışma, bulut tabanlı ve esnek modüler bir yapı aracılığıyla kullanıcı/sistem gereksinimi olan herhangi bir genişletmeye olanak tanımaktadır. Geleneksel monolitik tabanlı muadili ile karşılaştırıldığında, GTY 10 katın üzerinde daha iyi gecikme sonuçları sunar ve hizmet tıkanma oranını önemli ölçüde azaltır.

Anahtar Kelimeler: *Mikroservisler, ölçeklenebilirlik, modüler web uygulama, yönetim yazılımı, global ticaret yönetim sistemi*

1. Introduction

This paper introduces a professional web system & specific software application developed as a complete Global Trade - Import/Export Management (GTM) solution. Considering its integrated working with both formal organizations' systems & applications and intergovernmental infrastructures, it is highly required to be fully operable with regard to growing number of users and burst increase in transactions as well as any new expansion to the system functionalities. Based on a particular microservices architecture and serverless application approach together with a modular design strategy, the proposed GTM allows overcoming even unforeseen technical problems related to scalability and pertains to the ability of developers to maintain the whole application in the future. Its features provide high impacts on effectiveness, competence and quality.

One of the most challenging concerns in web application development nowadays is about smoothly handling the scalability issue while the number of users or load increases erratically. In fact, it is highly associated both with computer systems and business change with respect to the ability to adapt, especially in regard to persistent growth as well as increased demand. There are several ways to horizontally or vertically scale a web application system, suggested in the literature [1, 2]. Web developers prefer; for example, easing the server load, reducing the read load by adding redundant read replicas, reducing write requests, introducing more robust caching engine, offloading the database, minimizing the network traffic, moving static content to object-based storage, creating multiple availability zones or auto scaling the server.

Several contemporary studies on developing and deploying cloud applications based on microservices architecture aim at high scalability at the container level [3, 4]. Even without any domain knowledge, some favorable autoscaling approaches in cloud applications are proposed in parallel to technical progresses in reinforcement learning and machine

learning. In a similar design effort to build an intelligent autonomous autoscaling solution for microservices, quality of service (QoS) constraints are particularly considered for autoscaling in the cloud [5]. Whereby the proposed approach identifies the microservice resource demand by using a generic autoscaling algorithm running on the Google Kubernetes Engine and adapts the Kubernetes autoscaling paradigm considering the application resource requirements. After that reinforcement learning agents are used to learn and identify the autoscaling threshold values based on the resource demand and QoS.

As the main web system & product-based focus of this paper, a GTM solution addresses the requirements of national/international regulatory compliance, formal import & export processes, logistics management and financial information of an organization. GTM online web products aim at automating and simplifying the dedicated business processes in the sector as well as enabling businesses to minimize the total landed cost. Any GTM usage definitely results in improved productivity and visibility across the entire organization, allowing access to various supply chain mechanisms as well. It is all about contemporary and effective IT management as well as interworking and digitization of the several global trade-related processes. Compliance & duty management, import & export management, logistics management and analytics are the most important elements of a GTM with respect to the end-user considerations.

In the competitive landscape, there are several professional GTM software globally available. Proaktif (www.proaktif.org), Amber Road (www.amberroad.com), AEB (www.aeb.com), Oracle (www.oracle.com), Zonos (www.zonos.com), Infor (www.infor.com), SAP (www.sap.com), Aptean (www.aptean.com) and Descartes (www.descartes.com) are some of the well-known vendors. Although they aim to offer almost similar GTM services, there are certain differences with pros & cons in terms of system & software applications considering design, architecture and technical implementation points [6, 7].

The work presented in this paper has three major contributions. Firstly, a new contemporary microservices architecture-based GTM system & software is introduced. Secondly, it has been developed and implemented based on a modular approach to allow including and offering emergent internal or external services to be required by the users or customers. Finally, a serverless application strategy has been deployed in the design stages, granting the developers fully focus on writing the application and avoiding server management. The proposed GTM system & software with its microservices-based architecture not only provides more than 10 times better remote service delay results but also decreases the system blocking ratio less than 0.5% compared to its classical counterpart.

Rest of the paper is organized as follows. In Section 2, state of the art techniques has been explored for scalable web application development and microservices. Section 3 summarizes common GTM products with specific focus on web based solutions. The proposed GTM system & software based on microservices architecture and its basic performance evaluation is presented in Section 4. Section 5 concludes the paper with final remarks about scalable GTM or similar system development and suggests new directions into future researches.

2. Microservices architecture & serverless application approach for a new era in scalable web app development considering rigorous commercial use purposes

For the last decade, research about furious web application challenges has been focusing on technically efficient methods and their cost-effective implementations with regard to

ever-increasing user demands. Software developers have recently come up with new ideas in web system & application design based on microservices architecture and serverless application approach in order to dynamically handle both contemporary and foreseeable problems that are directly related to the scalability concerns.

2.1. Emerging scalability issues in web application deployment

Scalability problems usually occur in the GTM systems as the backend cannot scale at the same speed that the users surge into the web application. This is in nature related to the rise in the number of users and transactions between the external services as well as unpredictably increased and heavy interactions with the site at certain times.

A scalability and performance study aiming at reducing management and infrastructure costs for web applications in a cloud is presented in [1]. It introduces a dynamic scaling architecture based on a front-end load-balancer as well as a scaling algorithm for automated provisioning of virtual resources. Having analyzed on-demand capability of the cloud to promptly provision and fairly allocate required resources to users by means of a detailed experimental work, the proposed solution makes it is possible to achieve not only a sustainable performance upon abrupt load surges but also allocating system resources to the users adequately and utilizing the resources stably.

In another recent paper, a server implementation-critical question of vertical (up) or horizontal (out) scaling for linear and non-linear algorithms was addressed [2]. The former is based on increasing the number of enabled cores on a server node whereas the latter is based on increasing the number of server nodes basically. Indeed, the paper concludes with pros and cons of both approaches while running a mixed workload of linear and super-linear algorithms. Considering the underlying linear algorithms, if the provisions of parallelism are ignored while scaling-up, it is not possible to effectively utilize additional resources since it will continue running in a linear fashion. On the other hand, scaling-out the server by means of another parallel system is suggested as a feasible solution considering the case of underlying super-linear algorithms.

2.2. Eccentric solutions: Microservices architecture and serverless application

Compared to its predecessor monolithic applications, microservices-based solution offers great agility, high scalability, easy cloud-readiness, fast development cycles and platform/ language-free services. On the other hand, it has to deal with security, testing, monitoring and maintaining the network issues smoothly. Although there could be a few complexities with regard to best practices in structuring software in relatively small computation units (CUs) as depicted in Fig. 1, distinctly it allows highly optimized allocation of the web-app components within appropriate containers available through cloud service providers. The containers with several CUs in the virtual machines (VMs) working on the host machines (H) in this kind of infrastructures.

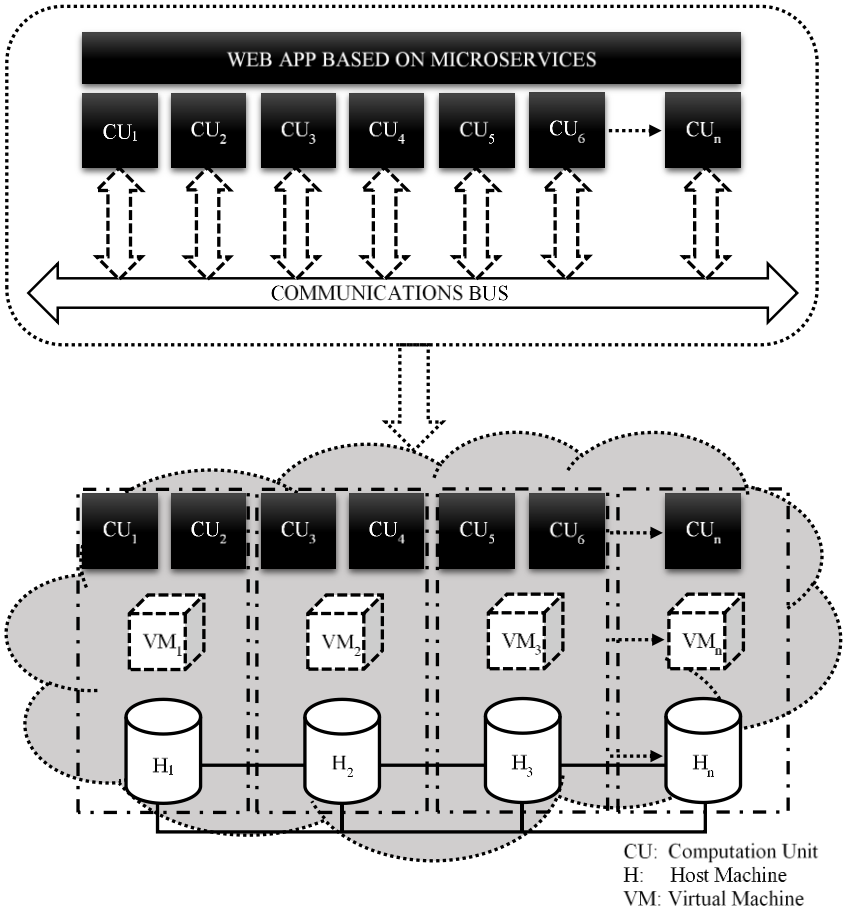

Figure 1. Software deployment in a cloud platform based on microservices.

In a contemporary study, both multi-version co-deployment and the corresponding version-oriented dependency management issues are addressed with regard to fundamental microservice frameworks [8]. It proposes a lightweight event-driven microservices (i.e., Spring Cloud) approach with adequate capability to quickly build applications connecting to external systems. It simply works at four stages. Version information from source codes of microservices are extracted, version dependencies are built, requisite versions of microservices are deployed just after packed and finally user requests are routed to desired versions at run-time. The work includes a detailed experimental analysis of the proposed framework implemented on Amazon Web Services (AWS) cloud environment, concluding that not only does it ensure the correctness of routing between multi-version microservices but also reduces the complexity.

Moreover, commonly used AWS cloud platforms are analyzed in [9], exposing both the main components with delay characteristics and crucial services playing a key role on latency sensitive & real-time applications. The paper proposes a detailed measurement methodology for Container as a Service (CaaS) or Function as a Service (FaaS) with regard to AWS platforms. Especially considering the delay characteristics of the AWS components, a comprehensive analysis is provided to be used to adjust a drone control application to the platform. Having investigated effects of the proposed approach on a contemporary system performance, the paper concludes that meticulously deciding the cloud services while designing the application that can tolerate about 100 ms delays, the AWS cloud approach can be utterly utilized.

Together with the use of microservices architecture in the GTM systems, serverless application strategy is also deployed to decrease costs for sporadic workloads as well as to overcome complex operational issues. Once implemented on AWS, it makes extensive use of backend as a service for latency-aware and extremely high level core functionality [10].

3. The proposed scalable GTM system & software based on microservices architecture

This section focuses on conceptual design and implementation of the proposed modular GTM system & software with major components and their roles based on a specific microservices architecture (Fig. 2). Considering its eleven main microservices, deciding how they efficiently communicate with each other is the first design challenge.

There are certain Application Programming Interface (API) gateways acting as entry points for interactions between the client user interface and developed microservices. They are highly incorporated with the load balancer and cache modules typically providing further performance improvement. Finding the right microservice is next step in order to forward user requests, for which a service registry mechanism acts as a Registration System (RS) for the developed microservices and their metadata. Both the API gateways and microservices consult the RS for services discovery as necessary.

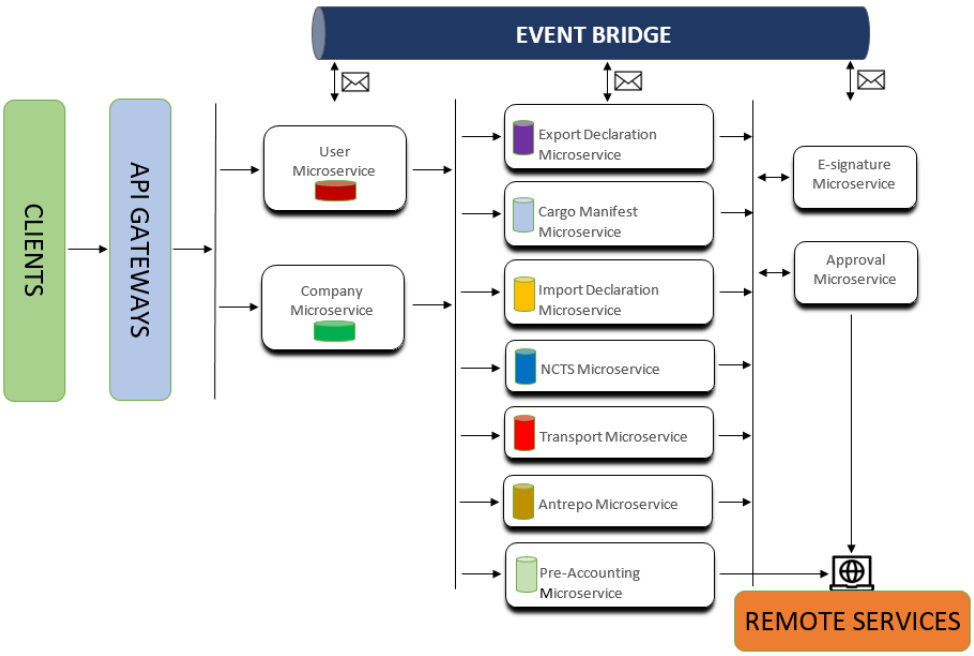

Figure 2. Proposed microservices arhitecture for the GTM system & software.

Security and authentication are also key implementation issues of the proposed GTM system & software development stages. They will ensure only authorized clients access to the business service among the several microservices as well as authentication between certain microservices. In addition, the proposed system essentially keeps the logs and monitors performance of the complete software and microservices based on runtime

statistics, user sessions and status of transactions. On collapse of any microservice, the saga (i.e., a sequence of local transactions) is considered to ensure the rollback of activities as a basic failure prevention solution.

3.1 The microservices

The proposed GTM solution includes User, Company, Export Declaration, Import Declaration, Cargo Manifest, New Computerized Transit System (NCTS), Transport, Entrepot, Pre-Accounting, E-signature and Approval microservices (Fig.2). By configuring the modules with microservices architecture that is superior to the classical monolithic approach, both technical and commercial performance of the GTM web/server applications have been raised to high levels with a flexible and expandable strategy. Carrying out the on-ground works or implementing the envisaged modular functionalities, they all execute either single logic or a single task sometimes involving thousands lines of code. Each microservice is meticulously built up in order to do only one task in order to eliminate any unnecessary communications with other ones, with respect to effective modularization strategy of the whole system design. As per the system requirements, the developed microservices with dedicated databases can highly benefit from the technology heterogeneity. Considering the professional functionality point of view, the microservices and services they provide the users & companies are succinctly listed below:

- Clients:
  - ✓ Devices that receive service from the proposed GTM system servers & software.
- API Gateways:
  - ✓ Intermediate layer used for clients to access the relevant service
  - ✓ Authentication and authorization management for accessing the services.
- Event Bridge:
  - ✓ Amazon service with real-time access to data changes in AWS services, the applications, and the proposed GTM software as a service (SaaS) application without writing code.
- User and Company Microservices:
  - ✓ User and authorization checks
  - ✓ Defining user roles for certain transactions
  - ✓ Admin definitions for the general system.
- Export and Import Declaration Microservices:
  - ✓ Import & export declarations and customs approval procedures
  - ✓ Declaration tax calculation, copying, Excel data transfer, XML data transfer, detailed report, collective declaration documents and declaration draft documents
  - ✓ Issuing circulation document from a declaration and approval procedures
  - ✓ Preparing electronic export invoice from a declaration
  - ✓ Union approval processes and payments
  - ✓ Informing the customers of the company users via e-mail
  - ✓ Electronic archiving

  - ✓ Creating invoices and receipts from a declaration.
- Cargo Manifest Microservice:
  - ✓ Marine import, marine export & all cargo manifest regime transactions and obtaining customs approvals
  - ✓ Copy of declaration, copy of deed, cargo manifest list
  - ✓ Cargo manifest draft documents
  - ✓ Electronic archiving.
- NCTS, Transport and Entrepot Microservices:
  - ✓ TR & T1 transit declarations and customs approval procedures
  - ✓ Online update of the status of declarations through the customs system
  - ✓ Creating automatic transit declaration from an import/export declaration
  - ✓ Electronic archiving
  - ✓ Guarantee calculation
  - ✓ Declaration draft documents
  - ✓ Entrepot declaration list
  - ✓ Entrepot reduction report and tracking of goods left in the Entrepot.
- Pre-Accounting Microservice:
  - ✓ Service invoice, debit receipt, receipt and bank document transactions
  - ✓ Company current follow-ups
  - ✓ E-invoice and e-archive transactions
  - ✓ Current transaction breakdown
  - ✓ Detailed reports (e.g., debtors, creditors and current document reports)
  - ✓ Electronic archiving.
- E-signature and Approval Microservices:
  - ✓ E-signature for required data
  - ✓ Communications to/from the remote services and approval.
- Remote Services (not limited to):
  - ✓ Ministry of Customs and Trade
  - ✓ Ministry of Finance
  - ✓ Exporters Assembly
  - ✓ Union of Chambers and Commodity Exchanges.

The user and company definitions are structured as the main framework. When a client wants to login to the application, first of all, user identification & authorization checks are made with the User Microservice. Secondly, the Company Microservice is used to define the users with authority to perform transactions on the certain companies. Finally, having successfully entered the application, the other microservices are made available to the users and companies for all customs, trade and transport related transactions. With services

such as Export Declaration and Cargo Manifest on the second level where all global trade transactions are made, functions like declaration writing, calculations, document printouts, document entry checks and ease of operation are then realized.

After entering the required data for a declaration, its approval must be obtained by communicating with different types of the Remote Services. Before this process, the prepared document or data is signed by using the Signature Microservice and then sent to the Approval Microservice that enables the signed file to be approved by communicating with the external service according to the service type (SOAP or FTP) to be used.

3.2 The development tools and implementation environment

The proposed GTM system and software has been designed and developed merely utilizing the AWS technology tools and infrastructure based on three-layered serverless architecture, namely presentation, logical and data layers.

At the presentation layer, both Amazon Simple Storage Service (S3) and Amazon Cognito have been used. The former allows a static website developed with Next.js to be served directly through an S3 folder without the need for a web server provisioning. The latter is a serverless user identity and data synchronization service that allows adding user registration, login and access control to web & mobile applications.

Amazon API Gateway, AWS Lambda, Amazon Simple Queue Service (SQS) and Amazon Simple Notification Service (SNS) have been used at the logical layer. All types of computing tasks, such as serving web pages, processing data streams, calling APIs and integration with other AWS services are performed with AWS Lambda functions. SQS allows developers to focus on different processes by eliminating the additional workload associated with the complex management and operation of messaging middleware, where messages can be sent, stored and received between all software components without any loss or further need for other services to be accessible. SNS is a fully managed messaging service for both application to application or person communication.

Considering the data layer, S3 and Amazon Aurora (MySQL) have been used for providing scalability, data availability, security & high performance as well as cost-effectiveness and simplicity of relational and open source databases with the availability of traditional databases.

In addition, Amazon Cloudwatch has been used for the traceability of all services, independent from the three layers. By using this service, a concrete system-wide visibility into resource usage, application performance and operational status is obtained.

4. Performance evaluation

An initial test version of the proposed GTM system & software application has been implemented first. It is specially used for the sake of performance analysis of the product as well as functional testing of the technical design challenges. As the performance metrics, this section examines normalized remote services delay (D) and system blocking ratio results obtained from the real experimental tests to assess scalability of the system & software.

While carrying out one week-time real experimental system and usage tests, daily volume of the data traffic (l) varies between 26.8 and 435.7 MByte/Day, and number of active users (n) varies between 42 and 481 (Table 1).

Table 1. Test data set

	Total Data Traffic Load, l (MByte/Day)	**Number of Active Users, n (Daily)**
1	26.8	42
2	52.8	114
3	131.3	442
4	135.7	454
5	155.6	466
6	173.4	474
7	435.7	481

Fig. 3 depicts the normalized remote services delay results against the increased load for the proposed microservices-based GTM system comparatively. Assuming an obtained experimental delay result for the classical monolithic-based GTM system (Dc) is 1 for a certain load (l), the corresponding obtained experimental delay result for the proposed microservices-based GTM system (Dp) is normalized as Dp/Dc.

A distinct improvement in the proposed microservices-based GTM system delay results can be clearly seen from the normalized graph as Dp is less than 10.9% that of the respective Dc for varying the l values. This is simply due to the fact that the proposed microservices-based GTM system remains functional in case of unexpected interrupts in the remote services (i.e., its other services can continue working properly) although still incurring increased delays whereas the classical monolithic-based GTM system fully fails together with its all other functional services during in a such a situation.

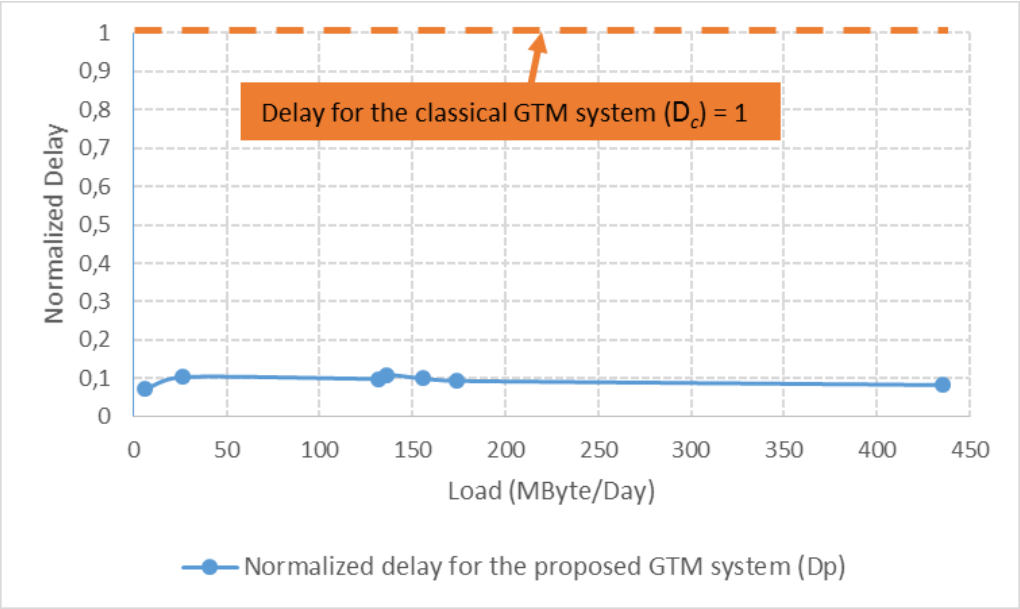

Figure 3. Normalized delay vs. load.

Fig. 4 presents the system blocking ratio results against the number of remote services usage (transactions). Increasing the daily total number of remote service usage up-to 20000, the proposed microservices-based GTM system well achieves full functionality while its classical monolithic-based counterpart suffers above 12.4% heavy failures. Considering its superior performance with regard to different types of interconnected remote services, the proposed microservices-based GTM system offers a seamless web application experience for the users.

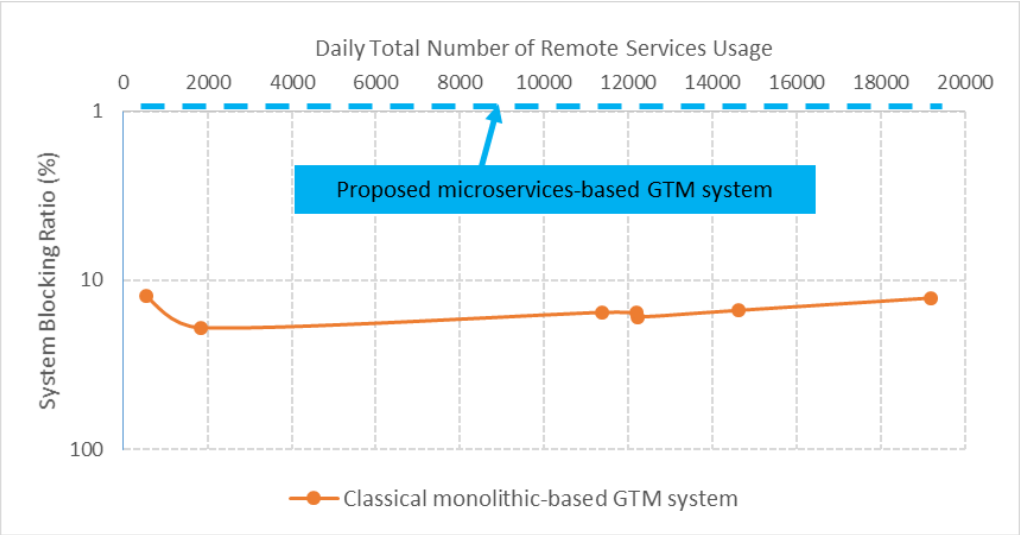

Figure 4. System blocking ratio vs. remote services usage.

In addition to its appealing technical and performance contributions, the proposed microservices-based GTM system allows a highly cost-effective solution bringing about up-to 45% decrease in server costs compared to the classical GTM system solutions as it deploys a serverless application approach.

5. Concluding remarks

This paper presents a microservices architecture and serverless application approach as a contemporary inception that is adapted to the proposed GTM system & software. Offering powerful advantages over its traditional monolithic architecture with respect to various user, system, software and service concerns, the project work carried out focuses on both conceptual design and practical implementation to build a professional solution in a specific industry.

The proposed GTM system & software with its microservices-based architecture is superior to the classical monolithic-based rival as it provides about 10 times better remote services delay performance. It also exceptionally decreases the system blocking ratio, which on the other hand dreadfully reaches up-to 19.04% in the classical monolithic-based solution.

Acknowledgment

The presented work is part of a R&D project financially supported by TÜBİTAK TEYDEB (Project Number: 7210001) and has been carried out at Proaktif Dijital Yönetim and Eğitim Sistemleri Ltd. Şti, Kocaeli University Technopark, Turkey.

References

[1] T. C. Chieu, A. Mohindra and A. A. Karve, "Scalability and performance of web applications in a compute cloud," **Proceedings of the 2011 IEEE 8th International Conference on e-Business Engineering, Beijing**, China, October 2011, pp. 317-323.

[2] S. M. Aaqib, "An efficient cluster-based approach for evaluating vertical and horizontal scalability of web servers using linear and non-linear workloads," **Proceedings of the 2019 3rd International Conference on Trends in Electronics and Informatics**, Tirunelveli, India, April 2019, pp. 287-291.

[3] S. Wang, Z. Ding and C. Jiang, "Elastic scheduling for microservice applications in clouds," **IEEE Transactions on Parallel and Distributed Systems**, vol. 32, no. 1, pp. 98-115, January 2021.

[4] I. Karabey Aksakalli, T. Celik, A. B. Can and B. Tekinerdogan, "Systematic approach for generation of feasible deployment alternatives for microservices," **IEEE Access**, vol. 9, pp. 29505-29529, 2021.

[5] A. A. Khaleq and I. Ra, "Intelligent autoscaling of microservices in the cloud for real-time applications," **IEEE Access**, vol. 9, pp. 35464- 35476, 2021.

[6] Z. Li-juan, "Research and application of SQL server in the trade management," **Proceedings of the 3rd International Conference on Computer Research and Development**, Shanghai, China, March 2011, pp. 209- 213.

[7] O. Pursky, A. Selivanova, O. Kharchenko, P. Demidov and V. Kulazhenko, "E-trade management system architecture," **Proceedings of the 2019 IEEE International Conference on Advanced Trends in Information Theory (ATIT)**, Kyiv, Ukraine, December 2019, pp. 283- 288.

[8] L. Liu, X. He, Z. Tu and Z. Wang, "MV4MS: A spring cloud based framework for the co-deployment of multi-version microservices," **Proceedings of the 2020 IEEE International Conference on Services Computing**, Beijing, China, November 2020, pp. 194- 201.

[9] I. Pelle, J. Czentye, J. Dóka and B. Sonkoly, "Towards latency sensitive cloud native applications: A performance study on AWS," **Proceedings of the 2019 IEEE 12th International Conference on Cloud Computing**, Milan, Italy, July 2019, pp. 272- 280.

[10] S. Eismann, J. Scheuner, E. V. Eyk, M. Schwinger, J. Grohmann, N. Herbst, C. L. Abad and A. Iosup, "Serverless applications: Why, when, and how?," **IEEE Software**, vol. 38, no. 1, pp. 32-39, January-February 2021.