IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 2, August 2023

Micro Services and its Interaction with API

Dr. Sandhya G[1], Dr. Manjunatha B N[2], Sudhanva J[3], Ankitha S[4], Navya K[5], Monika SP[6], Mohammed Numan Shariff[7]

School of Computing and Information Technology, REVA University, Bengaluru, India[1]
Associate Professor, Dept., of Computer Science & Engineering, R L Jalappa Institute of Technology, Doddaballapur[2]
Department of Information Science and Engineering, Nagarjuna College of Engineering Bengaluru, India[3]
sandhyag.pooja@gmail.com, manjunathabn@rljit.in

Abstract: *Microservices are compact programs with a single task that may be independently deployed, scaled and tested. The drawback of monolith architecture which is highly unstable and unreliable led to rise Microservice architecture. Microservices gives delivery mechanisms for service-oriented architecture*

Keywords: Microservices, SOA, API

I. INTRODUCTION

A service-based application development methodology is known as microservice. Big applications will be broken down into the smallest independent service units using this process. Microservices are a way of putting Service-oriented Architecture (SOA) into practice by breaking up an application into a number of interconnected services, eachof which caters to a single business need. Whole software packages will be split up into tiny, networked business units under a service-oriented architecture utilizing various protocols.

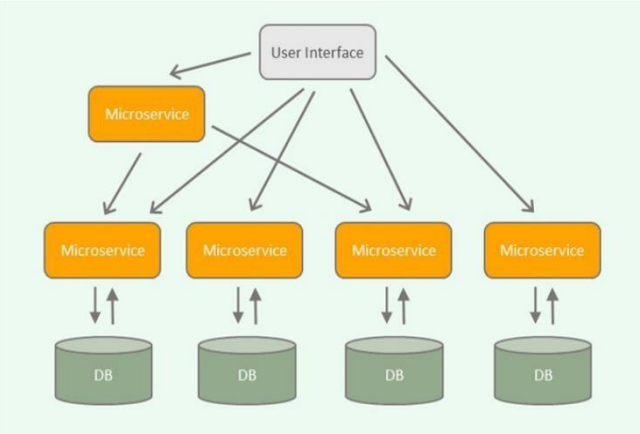

Fig 1. Microservice Architecture

II. LITERATURESURVEY

The cloud is upcoming paradigm that leads to many changes in the standards and methods . As per the needs of the cloud, architectural styles are also changing. In recent years microservices is seen as fast developing cloud application.
Microservices are a way of putting Service-oriented Architecture (SOA) into practice by breaking up an application into a number of interconnected services, each of whichcaterstoasinglebusinessneed.Themicroservicesare developed, deployed and maintained separately.[4] Whole software packages will be split up into tiny, networked business units under a service-oriented architecture utilizing variousprotocols.
Microservices gives delivery mechanisms for service oriented architecture. The advantages of microservices are modularity, continuous delivery and deployment, fault tolerance, improved scalability.

III. ALGORITHM

DFS PLACEMENT ALGORITHM: D-QSRFP
Input: Lu: $\{L(f_{i,j}) \mid \sum^{|N|} \lambda^{m}i, j > 0\}$

DOI: 10.48175/IJARSCT-12708

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 2, August 2023

Λ:{λ$^{i}_{i,j}$|∀$f_{i,j}$∈F,∀k∈N}N: serverlist
DG: dependency graph Output: Deployment Scheme 1: X ← **0**
2: λ_{solved} ← a dict whose values are 0 ∀s_i ∈ **S**
3: **while** len(L_u) ≠ 0 **do**
4: l_u ← maximum data transmission chain from L_u 5: coe ← 1
6: f_{prev}← None 7: **for** $f_{i,j}$**in** l_u **do**
8: **if** f_{prev}6≠ None **then**
9: coe ← coe × ACDC(fprev, $f_{i,j}$) 10: **end if**
11: fprev = $f_{i,j}$
12: λ_c = coe × $\sum^{|N|} \lambda^{m}_{i,j}$
13: λ_d = λ_{solved}[s_i] + λ_c − instNum(s_i) × μ_i14: **if** λ_d > 0 **then**
15: DeploySpread(s_i, [λ_d/ μ_i], X, N, DG) 16: λ_{solved}[s_i] ← λ_{solved}[s_i] + λ_c
17: **end if**
18: **end for**
19: L_u ← L_u \ {l_u} 20: **end while**
21: **return X**

IV. DISCUSSION AND RESULTS

4.1 COMPARISON OF MICROSERVICES WITH SOA

Both the designs allow for the creation of services using a variety of tools and programming languages, which diversifies the technology used by the development team. Multiple teams may arrange the work, but in SOA each team must be aware of the common communication channel. However, with microservices, each service may be deployed and run independently of the others. Therefore, it is simpler to scale a service independently or often deliver new versions of microservices.

In contrast to microservices, which aim to minimize sharing through "bounded context," SOA promotes the sharing of components. [4]A bounded context is one in which a component and its data are coupled as a single entity with few dependencies. Systems designed using SOA are likely to be slower than systems constructed using microservices, because SOA relies on several services to satisfy a business requirement.

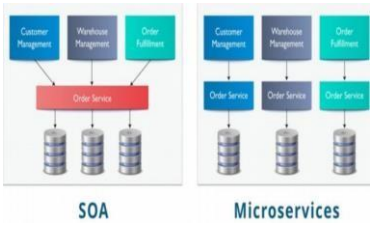

Fig 2 . SOA Vs Microservice Architecture

The Enterprise service bus (ESB) in SOA may develop into a single point of failure that affects the entire system.[3] Every service uses the ESB for communication, therefore if one of them becomes delayed, requests for that service may cause the ESB to become overloaded.

Microservices, on the other hand are significantly more error-tolerant. For instance, only that specific microservice will be impacted if there is a memory issue. The other microservice will all keep routinely responding to requests.

Fig3. Enterprise Service Bus for Traditional SOA

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 2, August 2023

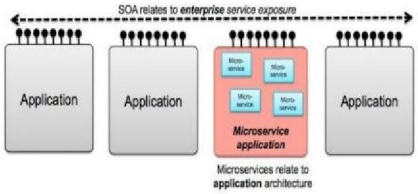

Fig4.Apllication Architecture

4.2 API INTERACTION WITH MICROSERVICES

There are two types of APIs when building system for communication in microservices architecture. Open/Public APIs that client applications can call. 2- Back end APIs that are utilized for back end microservice inter-service communication. In the case of public APIs, this should match client requests. In essence, synchronous communication is defined as using the Hypertext Transfer Protocol(HTTP) or Google remote Produce Call(GRPC) protocol to return real-time responses. Customer submits a request and waits for the service to respond. Therefore, the customer software pause its small set of independent instructions while waiting for the server to respond. The two protocols used for communication synchronously are HTTP and HTTPS. Data in HTTP is not secure as it is not encrypted where as HTTPS data is encrypted.

Asynchronous communication basically involves the entity user sending an appeal without waiting for a response from the service. The customer should not have stalled a thread while waiting for a reaction, which is the main point in this situation. AMQP(Advanced Message Queuing Protocol) is the most widely used contract for this asynchronous communication making RESTful HTTP APIs for microservices .[2]

When building a system for communication in synchronous mode, we should use RESTful (representational state transfer) when constructing APIs. The REST uses HTTP verbs which are usually called as method in nomad terms few of them are While on, POST, GET, and PUT are used to address the HTTP guidelines. While developing APIs for microservice communication, as it is most often used, we must make reference to the REST architectural method. The Sprint Boot framework in Java or the ASP. NET Core Web API services in C# are just a couple of the many solutions available to us for creating REST services.

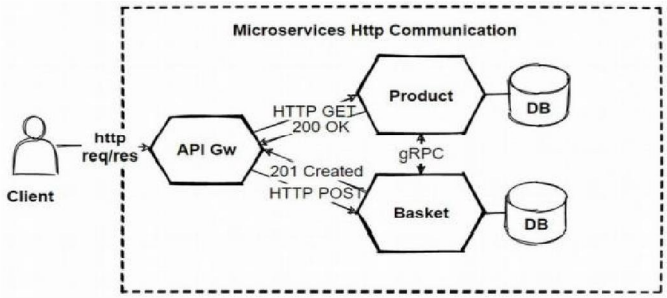

Fig 5. Http Communication

There are two different sorts of APIs used to construct sync communication in a microservices architecture. They are Public APIs, or APIs that client applications can call, and Back end APIs that are utilized for back-end micro service inter-service communication.[10]

The HTTP protocol request RESTAPI. The response can be of XML(Extensible Markup Language) or JSON(JavaScriptObject Notation).Most of the APIs reposes are in JSON. The GET-PUT-POST, DELETE, and other HTTP verbs provide the basis of API interface architecture functionalities communication in existing method since it is speedier as well as increases efficiency at transmitting as well as collecting data accompanied by reduced complications and reduced material than alternative formats.[7]Apps can connect with one another via REST by sending JSON data back and forth between the client and server.

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 2, August 2023

Fig6. Monolith Architecture- centralized Database REST shall be used at building APIs for solicit/reply

4.3 BENEFITS OF MICROSERVICES

Modularity

The main switch from traditional monolith/ SOA's to Microservices is because of the modularity. The splitting of the main application to the smaller modules enhanced the computation speed. The modules work independent and the services provided for each microservices are also independent.

Continuous Delivery And Deployment

DevOps philosophy cultivate continuous integration and continuous delivery. Microservices architecture allows multi-disciplinary teams to independently collaborate the task of develop, test, troubleshoot, , setup, and modify the services, which results in in quicker turnaround times for formation and regulate. The whole lifecycle of the formation process can be shortened by having the capacity to distribute the workload and make manual procedures automatic.

Fault Tolerance

Fault Tolerant Microservices are those that tolerate the fault. Furthermore, Micro service should be fault tolerant in order for the entire application to run smoothly. To implement this technique, "Resilience4j" library provides us with a number of modules based on the type of fault we want to tolerate.

Fig7. Advantages of Microservices

Improved Scalability

This capability is related to the scalability of microservices. As the workload expands with more and faster data, additional microservices can be deployed in parallel to spread the load across additional resources.

4.4 CHALLENGES IN MICROSERVICE

Increased Operational Complexity

Each microservice's team is usually in charge ofdeciding on and managing the technology to be used. Because each service must be deployed and operated independently, the cost and maintenance are increased. The Complexity increases as the number of Modules or Microservices increase.[1]

Less Communication

Microservices that are deployed independently function as miniature standalone applications that do not communicate with one another.

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 2, August 2023

Microservice Design

When the size of your application does not justify the need to divide it into numerous smaller components, a microservices framework may not be the best choice.[6] There is no need to further deconstruct applications that are already small enough.

Result

The DFS placement algorithm deploys the function chain with maximum data transmission size as depicted in line 4, and the services are deployed following the order of the function chain l_u. Then, the algorithm calculates the request rate for λ_c of each service in l_u and deploys the minimum number of instances to satisfy the desired ability λ_d in lines 8–17. Finally, it ends when all $f_{i,j}$that the users request are processed.

V. CONCLUSION AND FUTURESCOPE

The word "micro" in micro services alludes to the internal components' granularity, which means they must be much smaller than what SOA typically is. Within micro services, service components oftenserveaparticularpurpose and excel at it. On the other side, SOA services frequently implement as whole subsystems and typically feature a lot more business functionality.

Last but not least, the size and scope of a micro service versus a SOA are the primary differences. The micro service architecture is a type of SOA approach which comes through robust mechanism of enterprise level of architecture. The simplicity of modernizing the approach of SOA by higher order modularization is micro services. It is gaining momentum because of it scalable development and rapid deployment of software. Micro services will replace traditional software engineering in the upcoming years. Applications built using micro services have the qualities that are valued in environments that are flexible and business-focused.

REFERENCES

[1]. Raja Mubashir Munaf, Jawwad Ahmed, Faraz Khakwani and Tauseef Rana "Microservices Architecture: Challenges and Proposed Conceptual Design" 2019 International Conference on Communication Technologies (ComTech 2019)

[2]. Lorenzo De Lauretis " From MonolithicArchitecture to Microservices Architecture " 2019 IEEE International SymposiumonSoftwareReliabilityEngineeringWorkshops

[3]. Munaf, R. M., Ahmed, J., Khakwani, F., & Rana, T. (2019). Microservices Architecture: Challenges and Proposed Conceptual Design. 2019 International Conference on Communication Technologies (ComTech). doi:10.1109/comtech.2019.8737831

[4]. Ashraful Haque, Rasheq Rahman, Saifur Rahman " Microservice based Architecture of a Software as a Service (SaaS) Building Energy Management Platform " 2020 6th IEEE International Energy Conference (ENERGYCON)

[5]. Muhammad Waseem and Peng Liang,GastonMarquez,Amleto Di Salle " Testing Microservices Architecture-Based Applications: A Systematic Mapping Study " 2020 27th Asia-Pacific Software Engineering Conference(APSEC)

[6]. Guozhi Liu,BiHuang,ZhihongLiang,MinminQin,HuaZhou,Zhang Li "Microservices: architecture, container, and challenges" 2020 IEEE 20th International Conference on Software Quality, Reliability and Security Companion(QRS-C)

[7]. Francisco Ponce, Gaston Marquez, Hernan Astudillo "Migrating from monolithic architecture to microservices: A Rapid Review " 978-1- 7281-5613-2/19/$31.0 c

[8]. J. Abdul Rasheedh,Dr. S. Saradha," Review of micro-services architectures and IEEE Xplore Part Number:CFP20OSV-ART; ISBN: 978-1-7281-5464-0

[9]. IBM Cloud Education ,"Microservices"journal March (2021)

[10].Waseem,M.,Liang,P.,Marquez,G.,&Salle,A.D. (2020). Testing Microservices Architecture-Based Applications: A Systematic Mapping Study. 2020 27thAsia- Pacific Software Engineering Conference (APSEC). doi:10.1109/apsec51365.2020.0002