ADSOA Calculator

1st Ian Nikolay Doshner Galland
Information Technology Engineering
Universidad Panamericana
0219008@up.edu.mx

***Abstract*—The technologies of the 21st century are created with a centralized mindset and architecture, this causes that most of the problems arise from a hardware perspective lowering the systems or technology's availability. This project has both research and educational goals. The research objectives highlight the advantages in the area of distributed computing, including high availability, fault tolerance, autonomy and independence. Educational, provides a gateway to a new mindset in technology raising opportunities and new technological dilemmas, arising curiosity and inspiration to the students and readers with a code implementation of a distributed architecture made in different stages.**

***Index Terms*—Autonomy, Availability, Autonomous Decentralized Systems, Autonomous Decentralized Service Oriented Architecture, Distributed Computing, Fault tolerance, High reliability, Service Oriented Architecture.**

I. Introduction

This project has the creation and implementation of an Autonomous Decentralized Service Oriented Architecture (ADSOA) as its primary objective, this is obtained by coding a set of independent programs, them being a Client, Node, DataField, Cells and a Manager forming the ADSOA we desire. See Fig. 1. The structure of the paper is as follows: Sections II Through VI will explain and define the concepts used to clarify the scope of the project, Section VII will elaborate in how the definitions in the past sections are used in the project and finally the next sections will embark on the projects route, content, results and Conclusions.

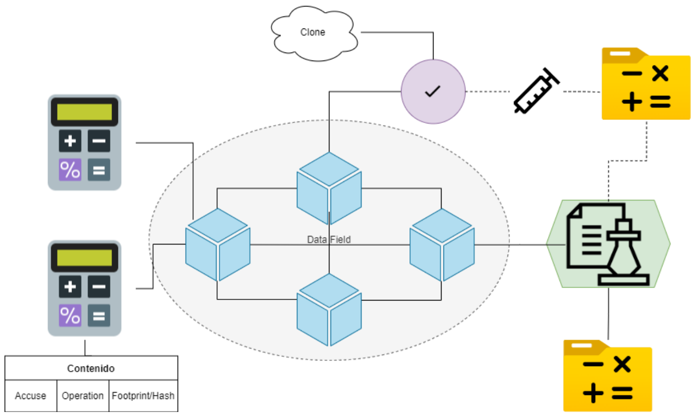


Fig. 1. Centralized System.

II. Centralized and Decentralized Systems

A. Centralized Systems

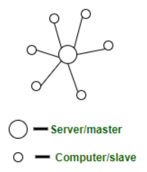


Fig. 2. Centralized System.

A centralized system is a system in which an individual, a group of people or a corporate entity holds the entire control over the functionality of the system.

Online social applications like Facebook, Twitter, Quora are examples of centralized systems.

We, end users don't have a say in the architectural design, feature availability or the functionality of the applications. We don't decide how the systems should operate. We have no control over the data. Corporate entities hold the rights to modify or delete our data without any permission.

A centralized system has its risks, for instance, single point of failure. If the company goes out of business all our data is gone, for ever. And this has happened in the past.

¿What is a Centralized Architecture or Centralized System?

Centralized systems are systems that use client/server architecture where one or more client nodes are directly connected to a central server. This is the most commonly used type of system in many organizations where a client sends a request to a company server and receives the response. Architecture of Centralized System: Client-Server architecture. The central node that serves the other nodes in the system is the server node and all the other nodes are the client nodes.

B. Decentralized System

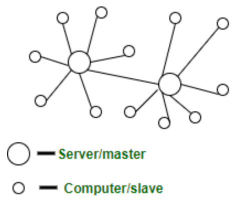


Fig. 3. Decentralized System.

These are other types of systems that have been gaining a lot of popularity, primarily because of the massive hype of Bitcoin. Now many organizations are trying to find the application of such systems.

In decentralized systems, every node makes its own decision. The final behavior of the system is the aggregate of the decisions of the individual nodes. Note that there is no single entity that receives and responds to the request.

III. Distributed systems

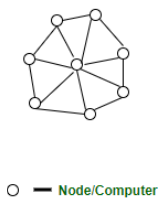


Fig. 4. Distributed System.

A distributed system is a set of computer programs that use computational resources on several different compute nodes to achieve a common shared goal. Also known as distributed computing or distributed databases, it relies on separate nodes to communicate and synchronize over a common network. These nodes typically represent separate physical hardware devices but can also represent separate software processes, or other recursive encapsulated systems. Distributed systems aim to remove bottlenecks or central points of failure from a system. Characteristics of distributed systems:

- Increased system reliability: this architecture has redundancy, when one of the systems fails, the others continue to work.
- Supportable growth: It is possible to make as many copies of the system in different new geographical locations.
- Local decision making: the business logic and decision making in each place are independent of each other.
- Data distribution: the data is not centralized. If the business logic consists of having local data in decision making, it is feasible, otherwise go to a centralized system.
- Use of local bandwidth: it allows to have a more friendly and attractive interface; only that it consumes the bandwidth of a local network
- Shared resources: Distributed systems can share hardware, software, or data
- Simultaneous processing: multiple machines can process the same function at the same time
- Scalability: compute and processing capacity can be scaled up as needed when additional machines are added
- Error detection: errors can be detected more easily
- Transparency: a node can access and communicate with other nodes in the system

¿How is our Project a Distributed System? Our project aims to establish an ADSOA Architecture by handling availability with an automatic system, package messaging and processing in independent systems. This will be done in 3 Breakthroughs shown in the following sections.

A. What is the difference between a centralized and a distributed system?

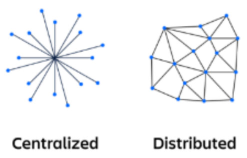


Fig. 5. Difference Between Distributed System and Centralized.

A centralized computing system is one in which all calculations are performed by a single computer at one location. The main difference between a centralized system and a distributed system is the communication pattern between the system nodes. The state of a centralized system is contained within a central node that is accessed by clients through a custom method. All nodes in a centralized system access the central node, which can overload and slow down the network. Centralized systems have a single point of failure. This is not the case with distributed systems.

IV. Service Oriented Architecture (SOA)

¿What is SOA? SOA is an infrastructure that separates system functions into individual components or services (e.g. access or security services, data collection, data dispatching, specific tasks...), which can be accessed over a network and used to develop applications by third parties. In this way, a collection of functions designed to reuse or access the system can be created to develop new applications and services by third parties. This system allows for faster development times, easier integration and greater functionality and standardization than the traditional time-consuming form of integral programming of all units.

In addition, through this architecture, services and devices can communicate more easily with each other, transmitting data and coordinating activities, which is one of the main objectives of IoT. SOA connects the different functional application units through well-defined interfaces and contracts between these services.

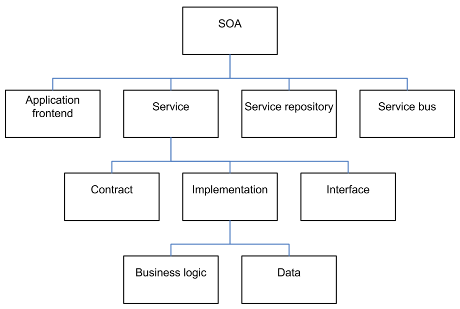


Fig. 6. Elements of SOA

A. *Microservices*

¿What is a Microservice? Microservices also known as the microservice architecture is an architectural style that structures an application as a collection of services that are:

- Highly maintainable and testable
- Loosely coupled
- Independently deployable
- Organized around business capabilities
- Owned by a small team

The microservice architecture enables the rapid, frequent and reliable delivery of large, complex applications. It also enables an organization to evolve its technology stack.

B. *Difference between distributed system and microservices.*

A microservices architecture is a type of distributed system because it breaks down an application into different components, or ”services.” For example, a microservices architecture might have services that correspond to business functions (payments, users, products, etc.), with each of the corresponding components handling the business logic for that responsibility. The system will then have multiple redundant copies of the services so that there is no central point of failure for a service.

V. Autonomous Decentralized System (ADS)

An autonomous decentralized system (or ADS) is a decentralized system composed of modules or components that are designed to operate independently but are capable of interacting with each other to meet the overall goal of the system. This design paradigm enables the system to continue to function in the event of component failures. It also enables maintenance and repair to be carried out while the system remains operational.

ADS architecture An ADS is a decoupled architecture where each component or subsystem communicates by message passing using shared data fields. A unique feature of the ADS is that there is no central operating system or coordinator. Instead each subsystem manages its own functionality and its coordination with other subsystems. When a subsystem needs to interact with other subsystems it broadcasts the shared data fields containing the request to all other subsystems. This broadcast does not include the identification or address of any other subsystem. Rather the other subsystems will, depending on their purpose and function, receive the broadcast message and make their own determination on what action (if any) to take.

VI. Autonomous Decentralized Service Oriented Architecture (ADSOA)

”Autonomous Decentralized Service Oriented Architecture is the culmination of joining ADS and SOA, to increase the availability of services and speed up the maintenance of the components in the systems.”[5]

VII. Implementation Goal

The objective of the project is the development of an ADSOA architecture, this will be divided into a number of clients that connect to a mesh of a number of nodes forming a data field, some cells where requests made by the client will be processed. Nodes can have clients connected to them as cells. Finally we will have an administrator that controls the injection of microservices to the cells that are up. All this will be developed in java with graphical interfaces made in JavaFx. There will be 3 Breakthroughs each one with something different and a new level of complexity Here are the representations of the components, the use of each component will vary in each breakthrough.

- Client

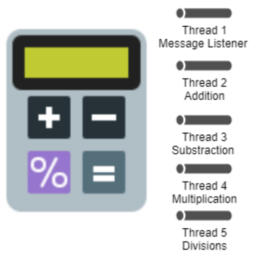


Fig. 7. Client

- Cell without Micro-services

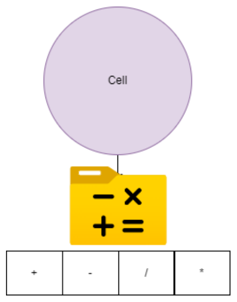


Fig. 8. Cell Used in B1

- Cell with Micro-services

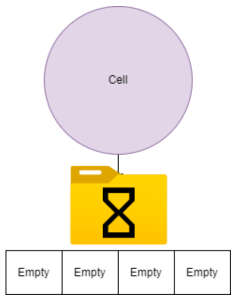


Fig. 9. Cell Used in B2 and B3

- Node

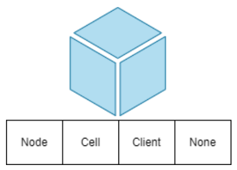


Fig. 10. Node

- Admin

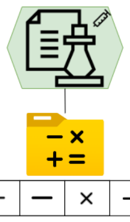


Fig. 11. Admin

VIII. First Breakthrough B1 (ADS)

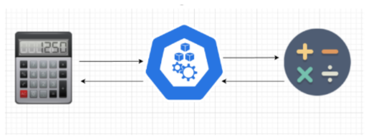


Fig. 12. First Breakthrough

Recreate the model shown made up of "n" clients, "n" nodes, "n" servers/Cells. This diagram 12 shows us a customer on the left side, a node/datafield in the middle and finally a cell/server on the right side. The first to get up must be the node/datafield and provide some channels to send and receive messages. From there the client or the cell/server can rise, but by recommendation we will always raise the cell / server first.

A. Project Objective in B1

- Create a front and back for the client that performs the functionality of a calculator
- Create a connection through sockets and sending Threads to the DataField/Node, the server/Cell and back.
- The messaging is done through Broadcast. That the operation sent by client "C1" is received by clients "Cn+1".
- That the operations are only performed in the cells/server.
- The client must be able to store the numbers and operations entered by the user and these send them following the model.

B. Project Components in B1

- Client/Calculator that sends petitions with a broadcast to all available nodes and displays the result in screen.
- Node that receives and sends packets through a broadcast
- Cell receives a package, process it and sends it to all available nodes through a broadcast. Shown in Fig. 8

C. Content Code (CC)

The format of the message sent was as follows:

- [Operation ID][Numbers before operator][Numbers after the operator]
- Operation IDs are cached and assigned as follows: 1 was a sum 2 a subtraction 3 multiplication 4 division.

IX. Second Breakthrough B2 (ASOA)

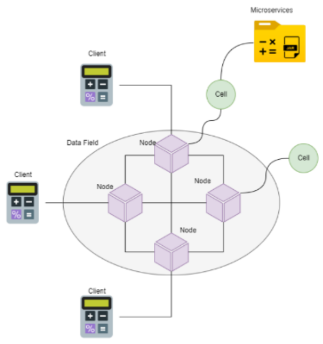


Fig. 13. Second Breakthrough

Once the basic functional infrastructure was obtained, the next level was to implement the service-oriented architecture (SOA) and a self-recovery protocol in case of failures.

A. Project Objective in B2

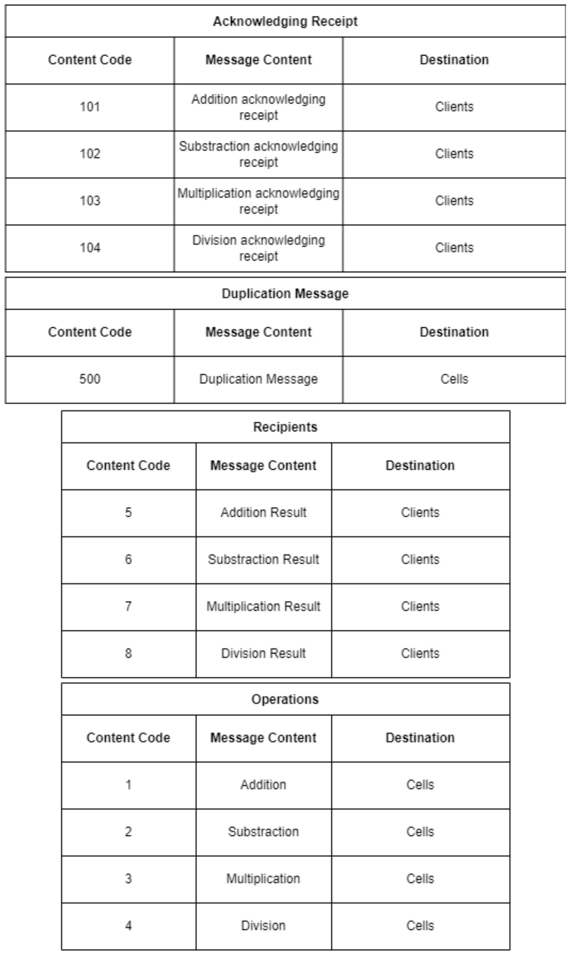
- The self-recovery protocol consists of the fact that the system needs at least 3 servers/Cells capable of operating all types of operations sent by the client. The client must wait until it receives 3 acknowledgments from different cells, this indicates that there are enough in the system for it to work correctly. If the client does not receive the minimum number of acknowledgments, it proceeds to try 2 more times. If it still does not receive a response, it sends a cloning message and the first cell to respond duplicates itself, notifying the others that it is no longer necessary. All the clients that are requesting the same kind of operation select the same Cell by alphabetically sorting the footprints and taking the first one, avoiding cancer. When an operation is unavailable all the other operations of the same type are held in a queue, while the client waits until the system has recovered. The other operations if there are enough Cells to perform them, keep working without any problem since each of them has a dedicated thread.
- The service-Oriented architecture consists of separating the operations from the cells. The operations passed from being a function inside the cells code to executable files that can be stored in a hard drive and, when the are needed, the cell loads them dynamically with java class loader.

B. *Project Components in B2*

- Client/Calculator that has the same functions as the B1 client but additionally sends a cloning message. Shown in Fig. 7
- Node that sends messages and plants a footprint. Shown in Fig. 10
- DataField A set of nodes and connections.
- Cell that is capable of loading dynamic classes and resolving client petitions. Shown in Fig. 9

C. *Content Code (CC)*

Now that the system grew and we added different types of protocols we needed to upgrade our CC, the first one was just an array with 3 elements and that was enough but now we have all types of codes, footprints, aknoledgments, etc. The next Figure will show the new content codes used in this Second Breakthrough.



Acknowledging Receipt		
Content Code	Message Content	Destination
101	Addition acknowledging receipt	Clients
102	Substraction acknowledging receipt	Clients
103	Multiplication acknowledging receipt	Clients
104	Division acknowledging receipt	Clients

Duplication Message		
Content Code	Message Content	Destination
500	Duplication Message	Cells

Recipients		
Content Code	Message Content	Destination
5	Addition Result	Clients
6	Substraction Result	Clients
7	Multiplication Result	Clients
8	Division Result	Clients

Operations		
Content Code	Message Content	Destination
1	Addition	Cells
2	Substraction	Cells
3	Multiplication	Cells
4	Division	Cells

Fig. 14. Content Codes B2

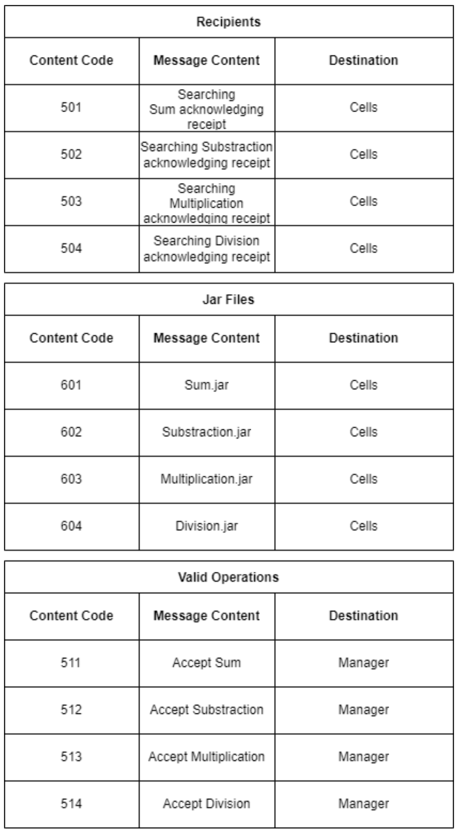
X. Final Breakthrough B3 (ADSOA)

In this final deliverable we still have the base developed in our 2 previous breakthroughs. Now we evolve by adding a new component, the Admin. This component is in charge of giving the user the option of selecting which operations the cells are capable of doing or not doing. To do this we must change the cells so that when a new cell is created it does not have the capability to do any operation. This is because now the operations folder inside the cell is going to be empty and until the admin says it will stay empty. Up next i will explain the objectives to accomplish this.

A. *Project Objective in B3*

- Cell behavior: We must now create each cell without an operation, that means that each cell must be empty at the beginning. Now since it does not count with the needed functions or jar files. The cell accepts the content codes for each type of operation but has an extra indicator, this indicator is set for it to know if it is capable of performing the operation or not before sending back a receipt. The cell can also accept content codes to indicate that they are capable of receiving a set of specific operations. (Only if they have this functions) and another content code indicating that the message contains the function?jar file for the operation.
- Admin behavior: The admin is an independent program that knows the location of the jar files and functions for the operations in the hard drive this is necessary for a functional injection. See Diagram shown in X-D The admin offers a simple menu where the user can easily interact, being able to send or inject the operation he wants in the cells. The Manager functions in the following way:
  - The manager detects the operation that is desired to inject
  - The manager sends a message with the code corresponding to ”searching for recipients” for the operations
  - The manager receives the response from available cells that can accept the operation. The CC has the footprints of the Cells.
  - The manager sorts the received footprints and starts injecting one by one. in a FIFO fashion.
  - The manager loads the file and sends it in a message.
  - After finishing the manager lets the user send a new operation.
- Cloning Cancer control Protocol: Previously when the client did not receive the minimum amount of receipts, it selected a Cell to clone itself, but now there might not be any cells capable of performing the operation, so no receipts are sent back. So we must protect our system from cancer, to do this when the client detects this behaviour it continues trying to obtain the operations result by re-sending it multiple times, but it does not ask any cell to clone itself, since there are no Cells available for that operation. See Diagram shown in X-D for more information.
- New Content Codes: Now that we have a complex network of connections we must ensure that each message in the system has a content code that indicates what information it is carrying. In the second breaktrhough we

saw a solution to this problem, where the content codes had messages for sending operations, results, receipts and cloning messages. Now we must change this. The next Content code shows the codes for searching for recipients for operations. When a Cell is not able to perform an operation because it does not have the necessary function, it is going to accept this ”searching for recipient” code for that operation. A cell that is able to accept an operation will send back a message containing the accepting operation content code for that operation and its footprint.



Recipients		
Content Code	Message Content	Destination
501	Searching Sum acknowledging receipt	Cells
502	Searching Substraction acknowledging receipt	Cells
503	Searching Multiplication acknowledging receipt	Cells
504	Searching Division acknowledging receipt	Cells

Jar Files		
Content Code	Message Content	Destination
601	Sum.jar	Cells
602	Substraction.jar	Cells
603	Multiplication.jar	Cells
604	Division.jar	Cells

Valid Operations		
Content Code	Message Content	Destination
511	Accept Sum	Manager
512	Accept Substraction	Manager
513	Accept Multiplication	Manager
514	Accept Division	Manager

Fig. 15. Content Codes B3

B. Project Components in B3

- Client/Calculator Shown in Fig. 7
- Node Shown in Fig. 10
- DataField
- Cell Shown in Fig. 9
- Clone Cell
- Admin/Manager Shown in Fig. 11

C. Microservice Injection

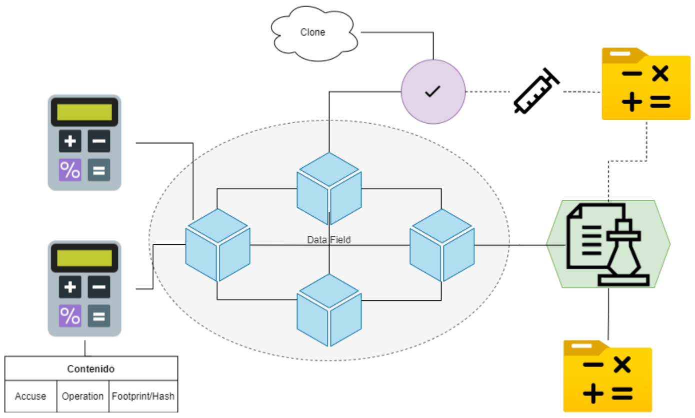


Fig. 16. Microservice Injection to a Cell.

D. Cancer Managment in Cell Cloning

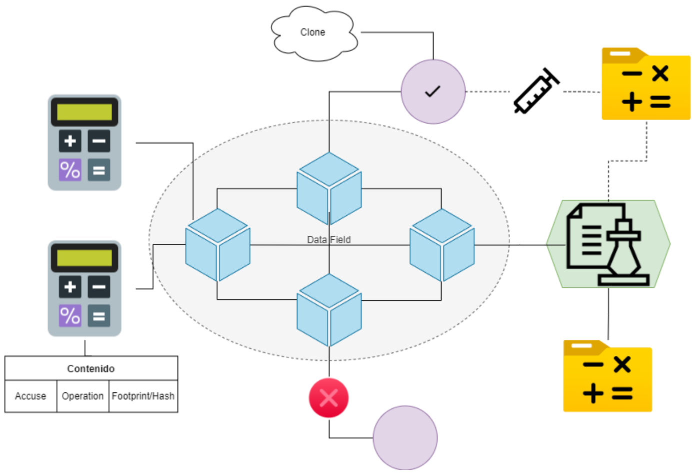


Fig. 17. Cell Cloning in case of a failure or petition management.

XI. Results

A. How to run the Code

`C:\Windows\system32` To run the code you must run each program and with a console window. You may run the program with the following commands `cd "Your Directory"` Type `java.exe -jar *filename*. jar.` You run n cells You run n nodes You run n clients You run an Admin and With that you have a replicated project mentioned in this paper.

B. Projects Code

You can find the source code to each independent component in the following github repository Click me!

XII. Conclusions

The document analyzed the application of the ADSOA architecture in a project. The next level of advancement of the distributed computing course and its transformation to ADSOA was shown. Distributed systems were found to be widely accepted and used in most modern software experiences. Social networking applications, video streaming services, and e-commerce sites are just a few of the technologies that use distributed systems. Centralized systems naturally evolve into distributed systems to manage scale. The use of microservices is a common and widespread practice to create distributed systems. The current status of the project and the possible future changes that the project may have were discussed. Thanks to the project, the advantages of distributed thinking and the ADSOA architecture can be exposed. Allowing the maintenance, update and repair of services without stopping the services raised. With this increasing the availability of not only the systems, quite possibly our future.

XIII. Front-End

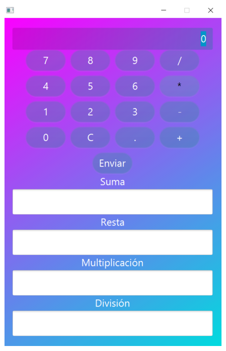


Fig. 18. Client Interface

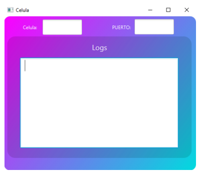


Fig. 19. Cell Interface

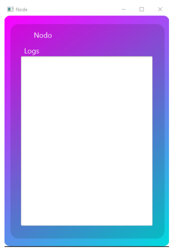


Fig. 20. Node Interface

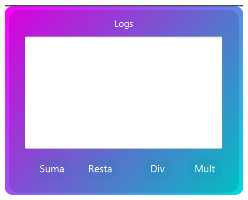


Fig. 21. Admin Interface

References

[1] "What Are Microservices?" microservices.io. Accessed November 24, 2022. https://microservices.io/.

[2] "Centralized Architecture in Distributed System." GeeksforGeeks, October 13, 2022. https://www.geeksforgeeks.org/centralized-architecture-in-distributed-system/.

[3] "Comparison - Centralized, Decentralized and Distributed Systems." GeeksforGeeks, November 9, 2022. https://www.geeksforgeeks.org/comparison-centralized-decentralized-and-distributed-systems/.

[4] Shivang, and ShivangHello World! I am Shivang. "Difference between Centralized, Decentralized amp; Distributed Systems Oversimplified." Scaleyourapp, August 9, 2022. https://scaleyourapp.com/difference-between-centralized-decentralized-distributed-systems-explained/.

[5] L. C. Coronado-García, J. A. González-Fuentes, P. J. Hernández-Torres and C. Pérez-Leguízamo, "An Autonomous Decentralized Service Oriented Architecture for High Reliable Service Provision," 2011 Tenth International Symposium on Autonomous Decentralized Systems, 2011, pp. 327-330, doi: 10.1109/ISADS.2011.49.