Proyecto “Calculadora con ADSOA”

Noguera Gil Andrei[#1]

[#]*Universidad Panamericana*
Augusto Rodin No. 498, CP 03920, Col. Insurgentes Mixcoac, Del. Benito Juárez,
Ciudad de México, México.
[1]0187940@up.edu.mx

Abstract— **Este documento es sobre la implementación de la arquitectura ADSOA para un servicio de operaciones básicas aritméticas, es decir, una calculadora que se conecta a una red de servidores para procesar la información y regresar un resultado.**

***Keywords*— ADSOA, Arquitectura de sistemas, calculadora, nodos, celulas, cliente, campo de datos.**

I. Introducción

Autonomous Decentralized Service Oriented Architecture (ADSOA) o arquitectura autónoma descentralizada orientada a servicios es la combinación de ADS con SOA que son plataformas que ofrecen una gran solución a sistemas, pero tienen elementos propensos a fallos y con un rendimiento inferior, por ello, al juntar estos dos sistemas se creó un nuevo panorama donde se reduce la posibilidad de fallos sin afectar la eficiencia del Sistema.

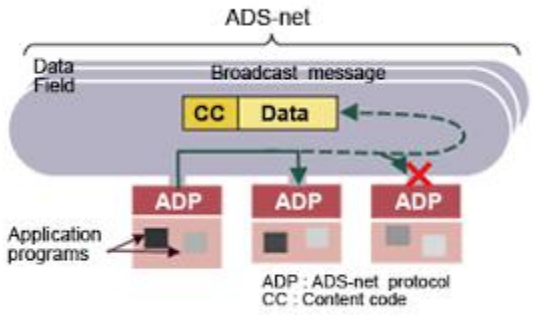


Fig. 1 Concepto de ADS

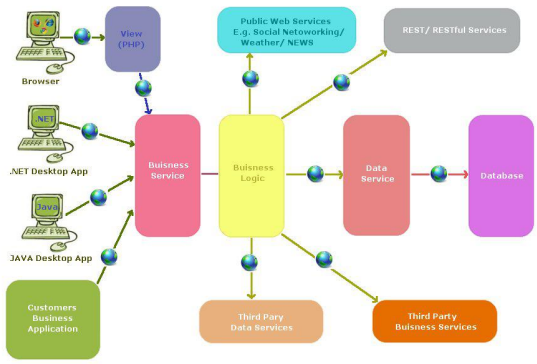


Fig. 2 Concepto de SOA

ADSOA tiene los beneficios que los dos sistemas ofrecen, el campo de datos y los servicios además de contar con elementos totalmente autónomos y descentralizados, es decir, que pueden subsistir sin depender de los otros elementos y pueden realizar sus propias acciones. La ventaja más grande que se obtiene es que permite que el sistema pueda reaccionar a fallos e incluso solo tener una caída parcial del sistema con un tiempo de recuperación bajo para recuperar el sistema lo antes posible.

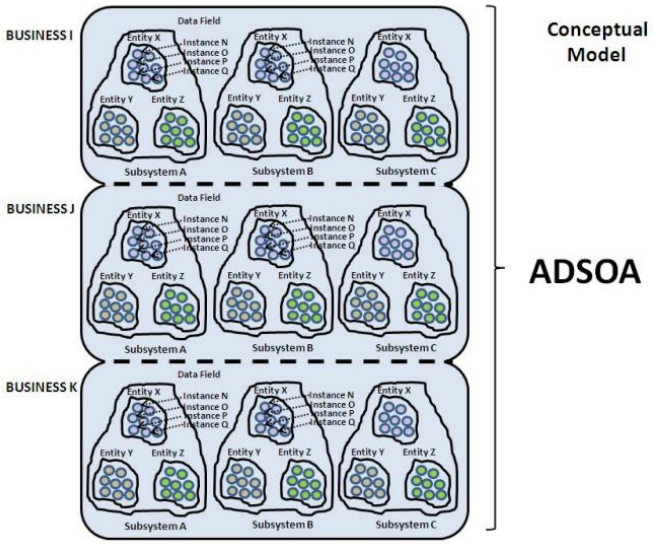


Fig. 3 Concepto de ADSOA

El funcionamiento de ADSOA es la segmentación de los sistemas hasta su mínima expresión y el conjunto de elementos resultantes generan subsistemas que a su vez generan subsistemas y así hasta lo posible, todos estos elementos resultantes están conectados entre sí sin importar la distancia o el medio todos tiene ser capaces de enviar al resto y recibir contenidos. Una analogía para entender este concepto, además de estar basado en este ejemplo, es el cuerpo humano. Cada hombre está constituido de un sistema de subsistemas, tenemos múltiples sistemas (Sistema respiratorio, sistema cardiaco, sistema central, entre otros) que tienen funciones diferentes, pero deben de trabajar en forma conjunta para que la persona viva, tomando un sistema (el sistema cardiaco) contiene varios órganos que son otros subsistemas, si nos concentramos en el corazón también tiene otros subsistemas,

al final llegamos a la parte fundamental y elemental de la vida, las células.

La teoría celular representa un sistema muy complejo que reacciona a fallos, tiene una comunicación muy efectiva donde los *"mensajes"* son recibidos y procesados por todos y solo aquellos que sean capaz de realizar una acción lo hacen y todos los demás lo desechan, tienen la capacidad de replicarse si el sistema necesita más células, pueden realizar cualquier función que el sistema necesite, al obtener una muestra de una región se puede determinar el estado del sistema, entre muchos beneficios más. ADSOA toma mucho en cuenta este tipo de funcionamiento y lo emplea en soluciones digitales obteniendo resultados muy prometedores. Una gran ventaja y desventaja es que al comportarse semejante al sistema humano tenemos las mismas probabilidades de presentar enfermedades o padecimientos como es el cáncer, las enfermedades o virus que afecte a todas las células, etc. La ventaja es que el ser humano ya ha aprendido sobre ellas tiene técnicas que también se pueden emplear de forma digital, sin embargo, la gran desventaja es que pueden desarrollarse incluso con los sistemas de detección más avanzados y por lo tanto puede ser inevitable la aparición de estos eventos.

La composición de entidades autónomas que ofrece o realiza peticiones de servicios por medio de mensajes, cada entidad debe ser única y contiene tres elementos para su conformar identidad: el id del subsistema, el id de sistema del negocio y el id de la entidad. Además, cada entidad está compuesta por varios elementos totalmente independientes y por lo tanto la conformación del sistema se representa por lo siguiente:

La composición de varias instancias o procesos constituye a una entidad, el conjunto de entidades constituye a un subsistema y el conjunto de subsistemas constituye a un negocio o sistema. Como se muestra en la fig. 3.

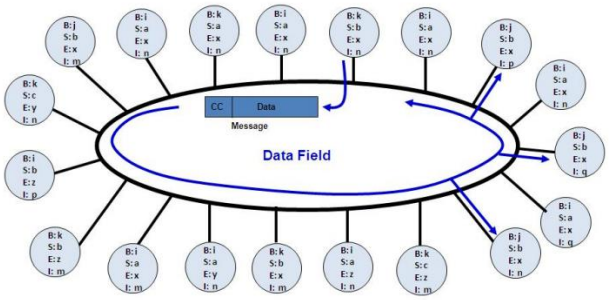


Fig. 4 Concepto de ADSOA

Se ha explicado el funcionamiento teórico del sistema y el funcionamiento práctico consiste en la implementación de un campo de datos (data field – DF) donde los miembros exteriores como puede ser un cliente manda una petición de servicio (un mensaje) al campo de datos, este se encarga de distribuirlos a todos, se procesa el mensaje y aquellos que tengan la capacidad de procesarlos mandarán una respuesta al campo de datos que a su vez la mandará de regreso al cliente.

El protocolo del campo de datos es:

I) La entidad pide permiso para ingresar al campo de datos
II) El campo de datos manda un mensaje
III) La entidad lo recibe y lo contesta
IV) El mensaje es como una prueba donde si la entidad lo responde correctamente puede ingresar al campo de datos

Por lo tanto, un sistema aplicado podrá tener la siguiente estructura:

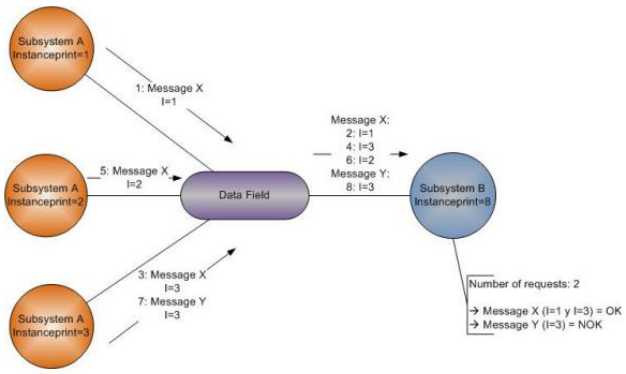


Fig. 5 Representación de un Sistema ADSOA

Donde en la parte central se encuentra el campo de datos, dos tipos de subsistemas conectados al DF que tienen funcionamientos diferentes. Supongamos que este sistema es una calculadora. El subsistema azul es la interfaz del cliente y los subsistemas amarillos son las operaciones que pueden realizar una calculadora (suma, resta, multiplicación y división) y que el primer subsistema puede realizar la suma/resta, el segundo puede realizar la multiplicación y el tercero la división. El cliente manda una petición al campo de datos y dependiendo de la operación se la manda solamente al subsistema correspondiente que puede procesar esa operación, una vez enviada el subsistema realiza dicha operación y se la manda al DF, procesa el mensaje y se la manda de regreso al cliente.

Este concepto básico de su funcionamiento representa beneficios muy importantes, en primer lugar, el campo de datos al ser totalmente independiente y autónomo garantiza la conexión y transmisión de mensajes, al tener separado las tipos de operaciones la eficiencia del sistema aumenta porque ya no se depende de un solo recurso además de poder

implementar varios subsistemas del mismo tipo en diferentes lugares, computadoras o instancias.

II. Previo desarollo

El proyecto consiste en la creación de una calculadora con una la arquitectura de ADOSA con la finalidad de familiarizarnos con la arquitectura y su implementación, además de obtener los beneficios que está ofrece en sistemas críticos en un nivel sencillo y práctico.

Para desarrollar este proyecto se dividieron en tres entregas correspondiente al periodo parcial de la institución académica (Universidad Panamericana). En la primera entrega debemos alcanzar el nivel mínimo para ADSOA que es la implementación del campo de datos, la conexión entre células por medio de los nodos para realizar peticiones de servicios que son las operaciones de la calculadora básica (suma, resta, multiplicación y división).

Los requerimientos son: desarrollarlo en un entorno de java, utilizando sockets para la conexión e hilos para la administración de procesos, la creación del campo de datos (son los nodos que forman el campo de datos y los encargados de transmisión de mensajes), la creación de una interfaz gráfica para la calculadora y donde el cliente puede realizar la solicitud de las operaciones.

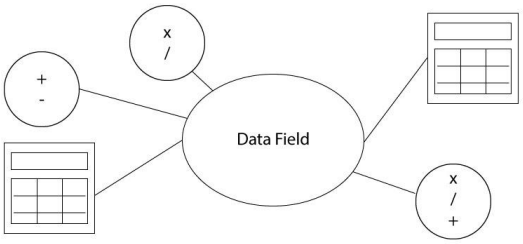


Fig. 6 Esquema del concepto de ADSOA

Campo de datos:

El protocolo del campo de datos para garantizar que todos los nodos están conectados realiza el siguiente proceso:

El primer nodo se instancia y empieza a realizar un escaneo a un determinado número de puertos, es decir, el administrador configura el número de puertos a escanear y si llega al último. pero no encontró ningún nodo el sistema levanta un nodo en la posición cero (La posición cero es el puerto 25000). Si encuentra un nodo disponible le envía un mensaje pidiendo la lista de los nodos que se han conectado a él. El sistema tiene ciertos inconvenientes, pero a su vez tiene muchas ventajas.

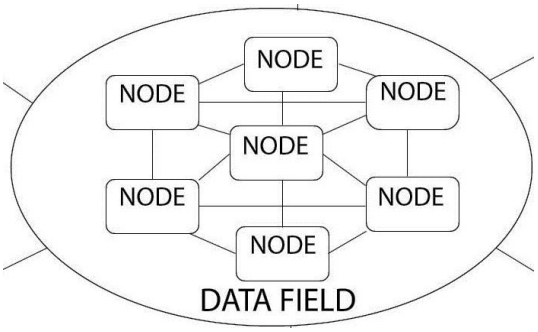


Fig. 7 Estructura del campo de datos

Los inconvenientes son que el sistema para su correcto funcionamiento debe de iniciarse un nodo y esperar a que se conecten a todos, el proceso que más tarda es el escaneo. La otra desventaja es que no se pueden iniciar dos nodos al mismo tiempo y el primer nodo siempre es el más tardado en levantarse. Sin embargo, el sistema ofrece más ventajas, una de ellas es que cada nodo almacena a los otros nodos conectados con él, puede compartir dicha lista para que al momento de recibirla pueda conectarse exitosamente con toda la red, el escaneo se vuelve casi nulo porque siempre tomará los nodos disponibles y si el primer nodo se cae el nodo que se levante tomará su lugar, cuenta con un sistema de monitoreo e identifica si un nodo se ha caído, además como todos comparten la misma lista se puede obtener que puertos están ocupando toda la red.

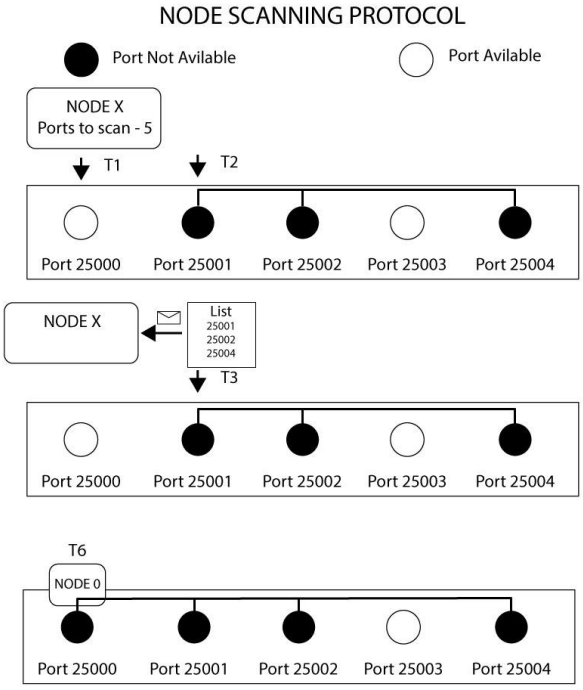


Fig. 8 Escaneo de nodos

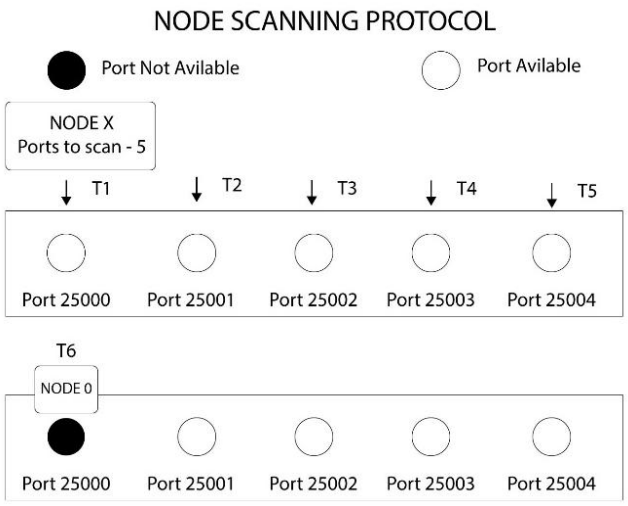


Fig. 9 Escaneo de nodos

Nodo:

Los nodos son la base del campo de datos donde contienen dos capacidades: La capacidad de permitir la conexión entre ellos y la capacidad de mandar los mensajes o recibir solicitudes.

El nodo tiene dos funciones, la primera es atender solicitudes locales donde solo involucran dos elementos, puede ser tanto un nodo con otro nodo o un nodo con una célula, pero no necesita ser enviado a los demás elementos. La segunda función es enviar los mensajes que no sean locales como son los servicios o mensajes de administración a todos los demás nodos y células. Por lo tanto, los nodos generan una red donde no procesan los mensajes si nos son directos al nodo (locales) y solo lo distribuyen.

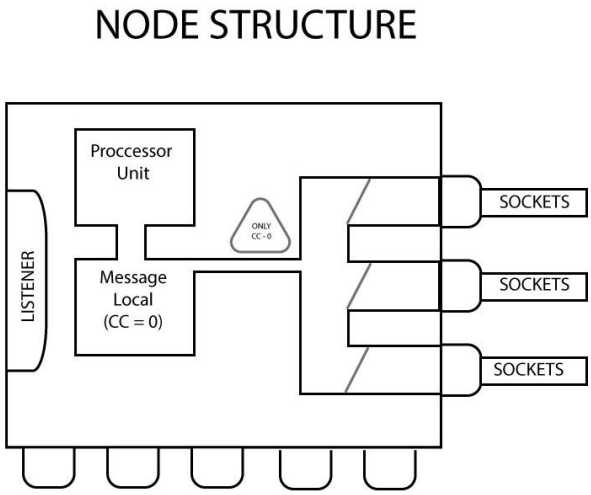


Fig. 10 Primera interfaz gráfica de la calculadora

La estructura de un nodo se puede visualizar como se establece en el esquema, donde tiene como función de servidor para poder levantar un servicio y aceptar las conexiones con los demás elementos y la de servidor que es la que tiene los sockets donde se reciben los mensajes. El procedimiento de análisis de mensajes es: Cuando recibe un mensaje solo corrobora si el Content Code (CC) es local (0) o no, si es local analiza el mensaje y procesa dicha solicitud dando como respuesta la lista de nodos, la conexión exitosa del nodo o un saludo, si es distinto no se analiza el mensaje y se distribuye a los demás sockets.

La sección de Listener tiene como función aceptar las peticiones de conexión y conectar el socket con el nodo para poder recibir y mandar mensajes. Como es una entidad lógica no tiene límite en aceptar las peticiones y la única limitante es los recursos que tiene la máquina para mantenerlos. Sin embargo, esto no representa un problema grave porque el sistema básico puede mantener una gran cantidad de elementos conectados a él.

Previamente se desarrolló la primera parte de la calculadora donde son componentes son:

- Data field
- Célula con la interfaz de la calculadora
- Célula con los servicios de las operaciones aritméticas

El data field es el encargado de mandar los mensajes a toda su red y a todas las celular conectadas, su protocolo es:

Mensajes de la célula con interfaz gráfica (Calculadora) catalogada como GUI manda el mensaje al Data Field donde los nodos solamente mandan el mensaje a las células con servicios catalogadas OPR. Por lo tanto, las células tienen un filtro que seleccionan a que célula debe llegar el mensaje. Si sale un mensaje de una célula GUI solamente tiene que llegar a las células OPR y viceversa.

Las células GUI tiene una interfaz gráfica muy sencilla y donde permite los mensajes que le llegan y los mensajes que se mandan.

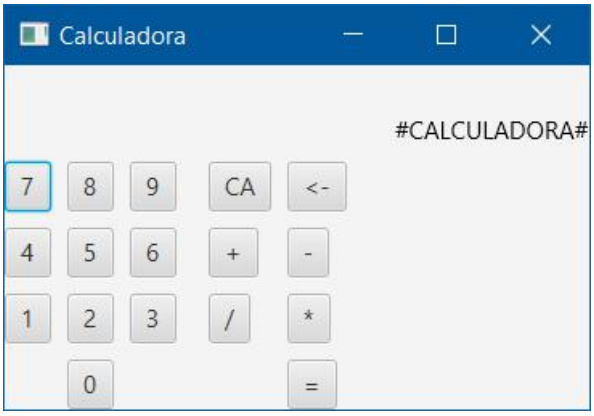


Fig. 11 Primera interfaz gráfica de la calculadora

Las células OPR son una consola donde notifica los mensajes recibidos y mandados, todos los servicios que proporciona (Suma, resta, multiplicación y división) están dentro de la célula y puede realizar las cuatro operaciones.

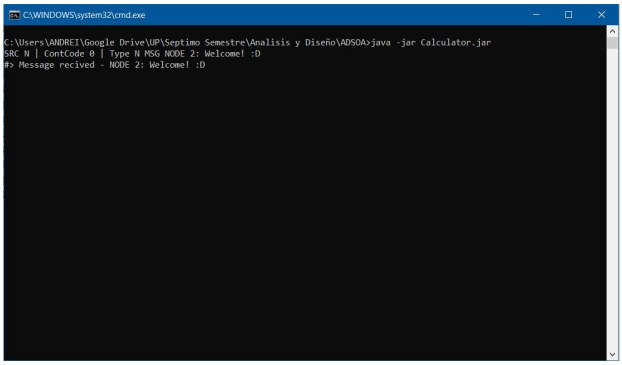


Fig. 12 Primera célula de servicios

Los mensajes tenían la siguiente estructura:

MESSAGES

CC	SRC	MSG	ACT	SND	IAR

Fig. 7 Primera interfaz gráfica de la calculadora

Donde:

CC: Content Code – Es un número entero que contiene el tipo de acción a realizar. Ej. Suma → CC = 5.

SRC: Source – Es la fuente de donde se manda, es un carácter donde *'N'* es un nodo, *'G'* es la interfaz gráfica y *'O'* es la célula de servicios.

MSG: Message – Es el mensaje mandado por el DF y es un string, las células interpretan y procesan el mensaje.

ACT: Action – Carácter que representa el tipo de acción a realizar, *'L'* es local y *'R'* es remoto. Si es remoto significa que viene de una célula y si el local significa que viene de uno de los hilos del nodo y se lo tiene que mandar la parte de servidor del nodo.

SND: Sender – Es el último elemento que lo mando, con el objetivo de garantizar que no se manden mensajes duplicados y generen un bucle de mensajes.

IAR: Integer Array – Arreglo de enteros donde se almacenan los puertos que están conectados, solamente aplica cuando a un nodo se solicitan los puertos.

Cada sección es muy importante para la transmisión de los mensajes. Los campos de CC, SRC, SND y ACT sirven para determinar el origen y destino del mensaje, el MSG y AIR sirven para poder mandar el cuerpo del mensaje, es decir, los mensajes consisten en 2 secciones: La primera es para determinar el destinatario y el emisor, la segunda es el contenido del mensaje.

Los mensajes son en formato estándar y todos los elementos del proyecto utilizan la misma librería del mensaje por lo que cada vez que se genera un mensaje se debe de crear un objeto de tipo mensaje y este objeto se manda a la red de nuestro sistema.

La lista de CC para los servicios son:

- 1 – Suma
- 2 – Resta
- 3 – Multiplicación
- 4 – División
- 10 – Resultado
- -1 – Lista de nodos conectados
- 0 – Mensaje simple [No requiere de servicio]
- 111 – Mensaje de supervisión

Para el segundo desarrollo del proyecto se deben de implementar la segmentación de servicios y la posibilidad de dar mayor disponibilidad y se pueden agrupar en las siguientes implementaciones:

- Colas de procesamiento
- Replicación / Clonación
- Servicio en Disco
- Archivo de configuración
- Acuses

También se cambió el comportamiento del DF para que ahora ya no haga la función de filtro y que sin importar el mensaje lo distribuya a todos los miembros conectados con la única excepción de filtro que si el CC es igual a cero significa que es un servicio local o que no es un servicio y que solamente es un mensaje de comunicación entre esa célula-nodo, si el CC es 1 es un servicio y eso significa un broadcast.

La estructura del mensaje también fue modificado para que sea más eficiente y se puedan mandar cualquier tipo de contenido.
Los mensajes tienen la siguiente estructura:

HEADER

CC	TS	FID	LS

MESSAGE CONTENT

OP	VAL	EVID

Fig. 13 Estructuras de los mensajes

Donde:

CC: Content Code – Es un número que indica si el mensaje es un servicio o no. 1> significa servicio y 0 significa mensaje local.

TS: Type Sender – Número que representa el tipo de elemento que manda el mensaje, cualquier célula representa un 1 y un nodo representado por un 0.

FID: Finguerprint ID (Huella) – Es el identificador único de cada célula que permite identificar exactamente de qué célula mandó el mensaje.

LS: Last Sender –Contiene el tipo (Célula o Nodo) de elemento que manda el mensaje, es decir, el elemento que está mandando el mensaje con el objetivo de conocer si el que recibió el mensaje es un nodo y lo está mandando de nuevo o lo está mandando una célula directamente. Es útil para evitar un bucle del mismo mensaje en la red.

OP: Operator – Número que representa el tipo de operación o de servicio a realizar y junto con el CC se puede ocupar el mismo número, pero con acciones diferentes, es decir, si el OP igual a 5 es sumar en un servicio éste mismo número con una operación de no servicio puede significar otra operación como mandar lista de nodos.

VAL: Values – Es un arreglo dinámico que no tiene ningún tipo de variable establecida por lo que se puede mandar cualquier tipo de contenido sin la necesidad de mandarlo todo por string o una variable en específica.

EVID: Event ID – Almacena la cadena del evento de la operación, este campo está diseñado para almacenar el evento de la operación y con ello podemos saber que es lo que hace sin necesariamente ver el contenido del mensaje, solamente esto no es tan notorio para los servicios porque cada uno son único.

Se divide la estructura en dos secciones, una de *"Header"* que contiene todos los datos para enviar el mensaje y la otra sección contiene el mensaje y las operaciones que debe de realizar.

Al mandar un mensaje a través del DF el CC y LS determinan si se debe de mandar a todos y si deben mandarlo a las células y nodos o solamente a las células. El TS y FID permite al usuario poder rastrear el mensaje y dar información valiosa sobre el mensaje como reconocer de donde proviene. El OP significa la operación que debe de realizar y en caso de las células si reciben un servicio que no pueden proporcionar lo desecha o si necesitan realizar una clonación toma una acción distinta. El VAL es muy útil para poder mandar mensaje porque dependiendo de operación solicitada se puede castear a las variables que necesita y poder procesarlos, y el EVID sirve para poder identificar específicamente un proceso porque se generan de manera única.

Colas:

Las colas de procesamiento consisten en 3 colas, una cola de para los mensajes entrantes, una para los mensajes en proceso y una para los mensajes salientes.

La cola de mensajes salientes en la encargada de recibir el mensaje, almacenarlos para crear un stack de mensajes y no dejar de recibir ninguno, después cada mensaje se manda a la cola de procesamiento.

En la cola de procesamiento dependiendo del servicio o proceso a realizar se genera una acción, si el proceso es un acuse lo almacena en un arreglo de acuses, si recibe una respuesta manda un acuse de recibido, si es una operación lo manda el mensaje al campo de datos, almacena el mensaje en un arreglo de operaciones en proceso y lo manda a la cola de mensajes salientes en espera de todos los acuses.

En la cola de mensajes salientes se guardan las respuestas y junto con las operaciones mandadas con el objetivo de seguir almacenadas hasta que se reciben los acuses necesarios, si se reciben los acuses necesarios se almacenan en un arreglo de procesos completados y se eliminan de esta cola. Si se reciben más respuestas de un proceso completo se eliminan. La única condición que esta cola no esté vacía es porque todavía existen operaciones por realizar.

Existen tres arreglos para poder controlar los distintitos procesos y las colas, un arreglo para almacenar los acuses y de ahí se guarda todo el mensaje para confirmar el origen (para evitar acuses duplicados), el evento (que determina la operación), el folio generado por ese acuse y los demás datos del mensaje como el último elemento que lo mando junto con la demás metadata. Otro arreglo para los procesos inconclusos donde se almacenan todos los procesos que aún no se han completado y solo se almacena el evento asociado. El último almacena los procesos ya completados y también se almacena solamente los eventos.

Estos tres arreglos junto con las colas son los que controlan el flujo y acciones de los mensajes para su correcto funcionamiento.

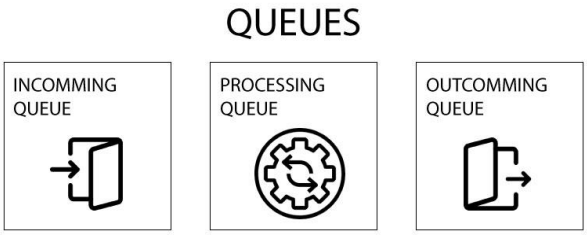


Fig. 14 Esquema del funcionamiento de las colas

Servicios en disco:

Los servicios en primera instancia existían en la célula de servicios y podía realizar todos los servicios, por lo tanto, todas las células podían realizar cualquier operación y no necesitaban de un evento, ahora los servicios se almacenan en disco donde solamente existen las operaciones que puede realizar.

Al llegar un servicio la célula ejecuta el programa en disco, le pasa los parámetros del método, ejecuta el método y regresa el resultado, al final se destruye el programa y manda el mensaje de resultado.

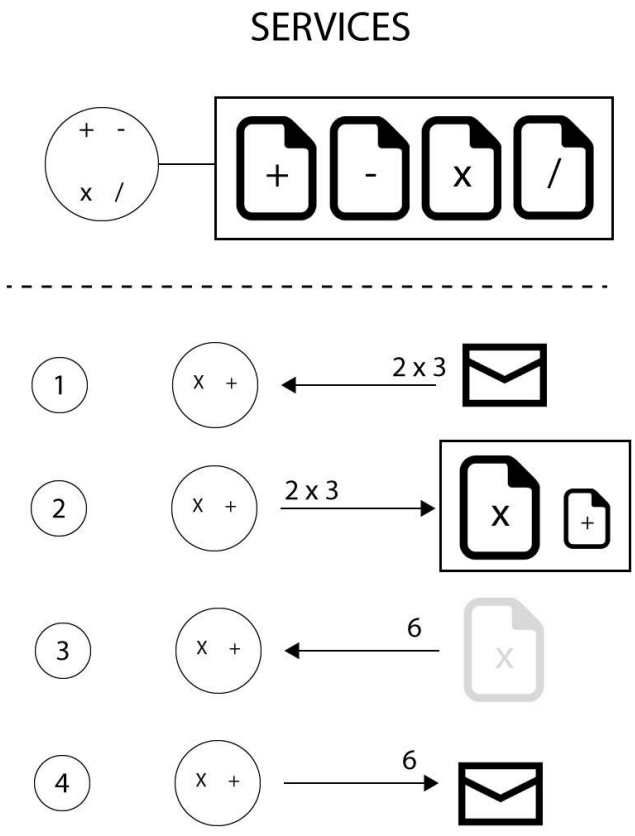


Fig. 15 Archivos de servicios

Archivo de configuración:

Para la configuración de las células de servicios se realiza por medio de archivos de configuración, en este archivo solamente contiene los servicios que puede ejercer las células y en el solamente tiene los números de los servicios. Los // escritos en archivo son considerados como comentarios.

El archivo de configuración contiene la siguiente estructura:

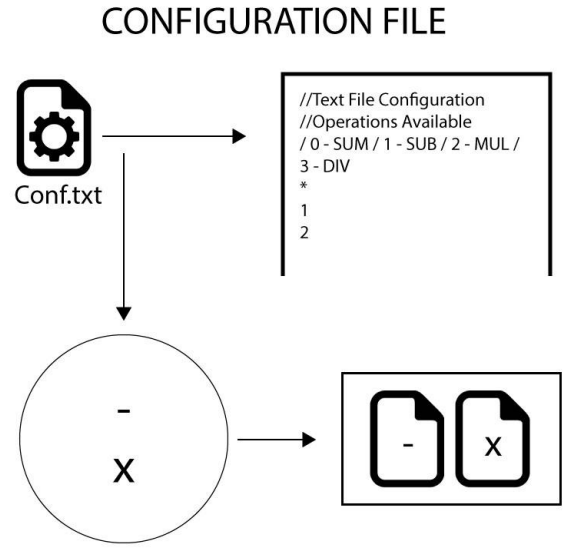


Fig. 16 Archivos de configuración

También para garantizar que las células no puedan realizar más que los servicios proporcionados por el archivo de configuración en la carpeta de servicios solamente existen dichos archivos.

Acuses:

Los acuses son los folios de recibido que se mandan al recibir una solicitud de un servicio o al recibir un resultado de un servicio, este tipo de mensajes sirve para poder determinar si un mensaje ha llegado correctamente o ha llegado al destino y en caso de no llegar tiene que enviar de nuevo ese mensaje y hasta no recibir los acuses necesarios no puede terminar o completar dicho proceso.

Los acuses se genera un folio único, pero siempre se le asigna el ID del evento de la operación para identificar el mensaje que recibió.

Las funciones de los acuses son:

- Evitar la pérdida de mensajes
- Clonación
- Notificación de baja disponibilidad de un servicio

Y todas las células deben ser capaz de mandar acuses y procesar los acuses.

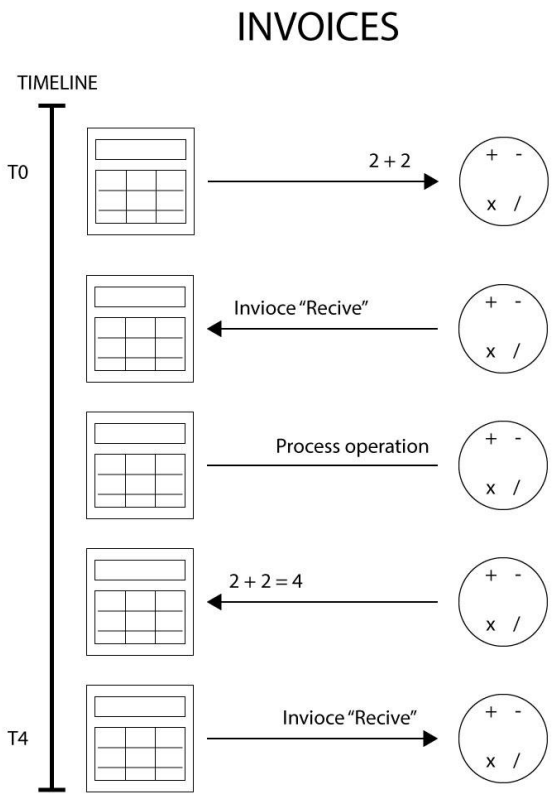


Fig. 17 Ejemplo de acuses en una interfaz

Clonación:

La clonación consiste en replicar una célula de operaciones con el objetivo de siempre tener una alta disponibilidad de los servicios. Cuando la célula se replica debe de ser capaz de realizar las mismas configuraciones que la original, es decir, debe de tener los mismos atributos que tiene la original sin la necesidad de levantarlas manualmente.

Para ello cada interfaz gráfica tiene sus configuraciones de cada operación, estas configuraciones se determinan por el nivel de criticidad de cada servicio y al momento de detectar una baja disponibilidad tiene que lanzar una alerta para solicitar que una célula clone sus servicios (a si misma).

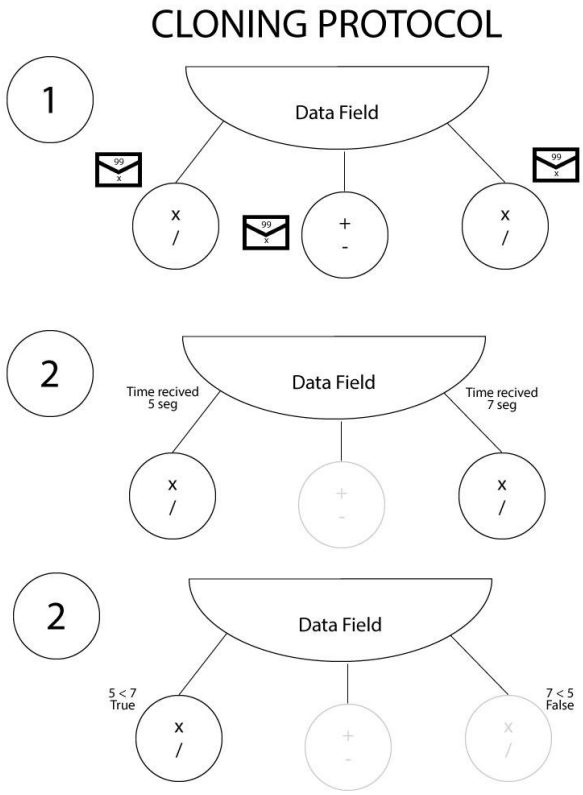


Fig. 18 Protocolo de clonación

Se creó un protocolo de clonación para garantizar que pueda generar una sana clonación de las células y evitar la clonación desmedida, el protocolo consiste en: Al recibir un mensaje de clonación que contiene [CC = 1 – Servicio | OP = 99] las células de servicios entran aun estado de clonación donde todas ellas se activan para clonarse, mandan un mensaje al DF mandando el tiempo de cuando recibieron el mensaje, las demás células reciben el mensaje y lo comparan con el tiempo donde ellos lo recibieron, si es menor siguen con la clonación y si el tiempo es mayor se cancela la clonación. Después de 3 segundos si la célula de menor tiempo no recibe algún tiempo menor empieza con la clonación. Al siempre tener que esperar un tiempo menor garantiza que solamente una célula se clone.

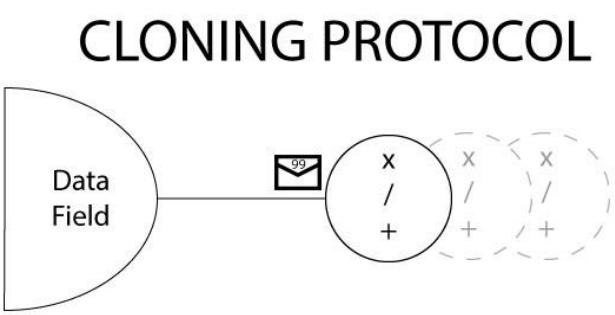


Fig. 19 Clonación

III. Desarrollo de la calculadora

Como último desarrollo del proyecto se incorpora el desarrollo de un manejador de células que permite a las células de operaciones cambiar su configuración o ADN, lo hace por medio de mensajes al data field donde las células reciben esa solicitud y, si cumplen con las condiciones, agregan dicha la operación enviada. Esta propiedad solamente puede ser generada por este manejador que tiene las operaciones en disco y solamente por medio de sus mensajes se pueden hacer las modificaciones.

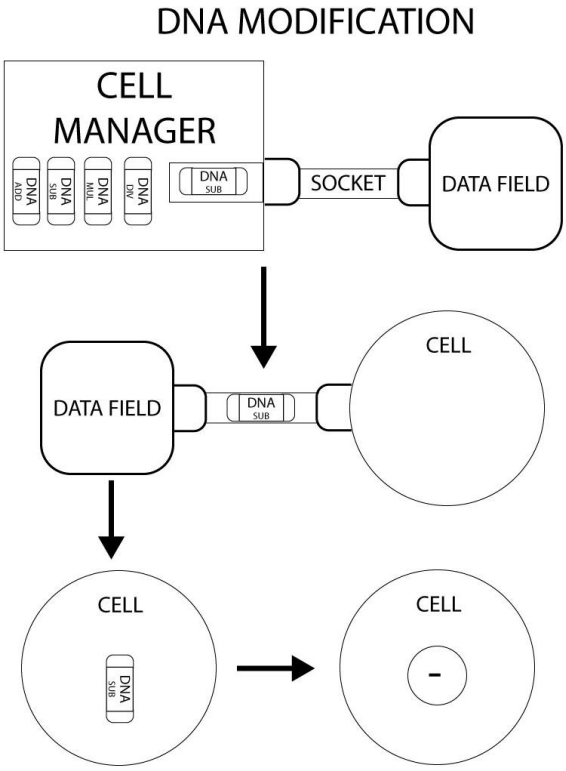


Fig. 20 Inyección de ADN

También existe la función del manejador de células para remover un servicio, para ello manda un mensaje de remover junto con las condiciones en específico para que las células que lo cumplan con las condiciones remuevan el servicio solicitado.

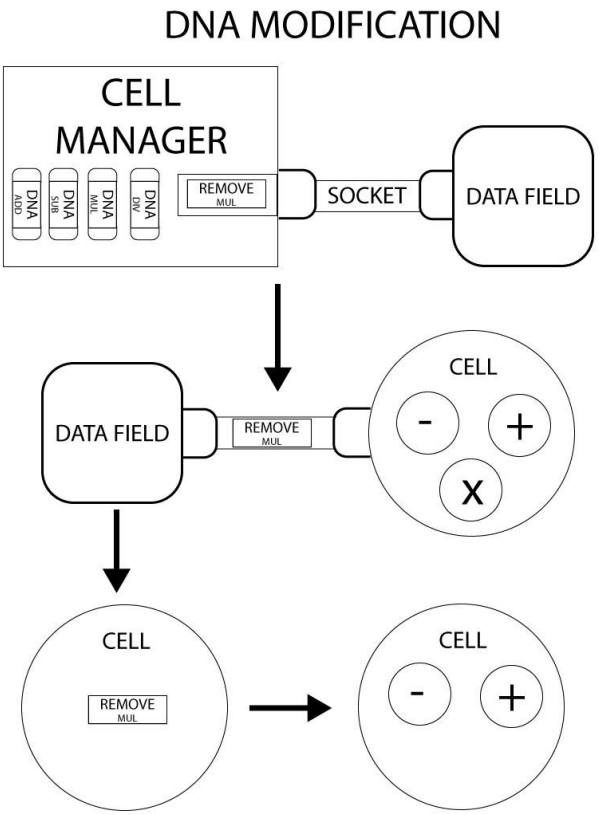


Fig. 21 Mensaje de remoción de un servicio

Por último, aparece el concepto de célula madre donde es una célula que tiene la capacidad de obtener cualquier servicio.

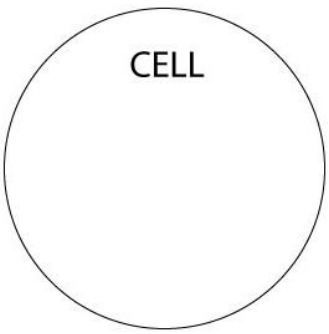
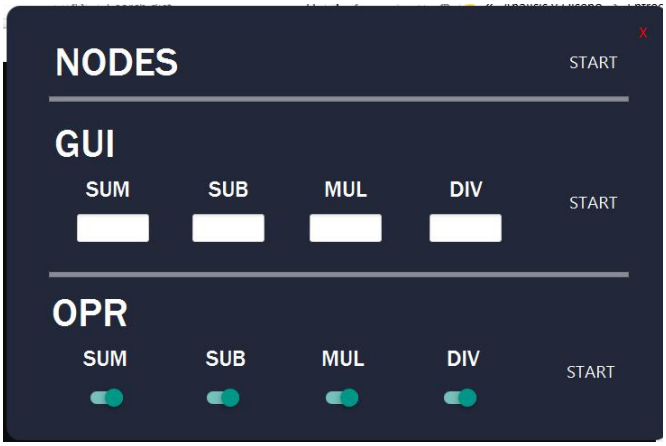


Fig. 22 Concepto de célula madre

Implementaciones Adicionales:

Para poder administrar las instancias se realizó un controlador que permite poder instanciarlas, pero no controlarlas y tiene la siguiente vista.



Los protocoles que permiten a la calculadora permitir servicios, fragmentar servicios, tener una alta disponibilidad, establecer un nivel de criticidad, entre otras cosas son los implementados en la segunda entrega. Y gracias a todas las implementaciones no permiten tener un sistema más sólido y robusto que permiten tener varios clientes (interfaces) y varios servidores (Células de servicios) en un mismo sistema con un funcionamiento correcto, es decir, que no importa los números de interfaces, células de servicios y nodos que contengan el sistema las peticiones siempre van a llegar a las células correspondientes y los resultados a las interfaces correctas quitando esa limitación del primer modelo.

El sistema ya se puede representar de la siguiente manera:

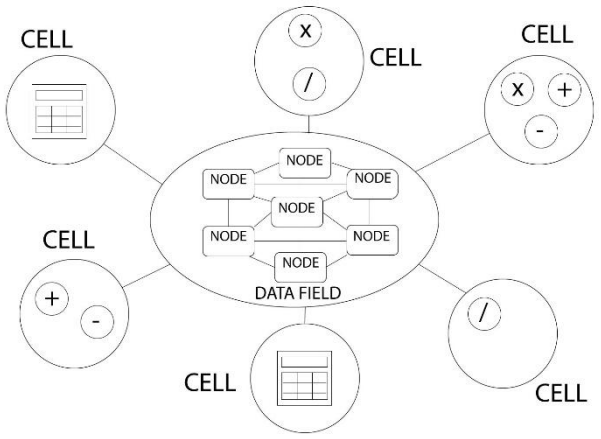
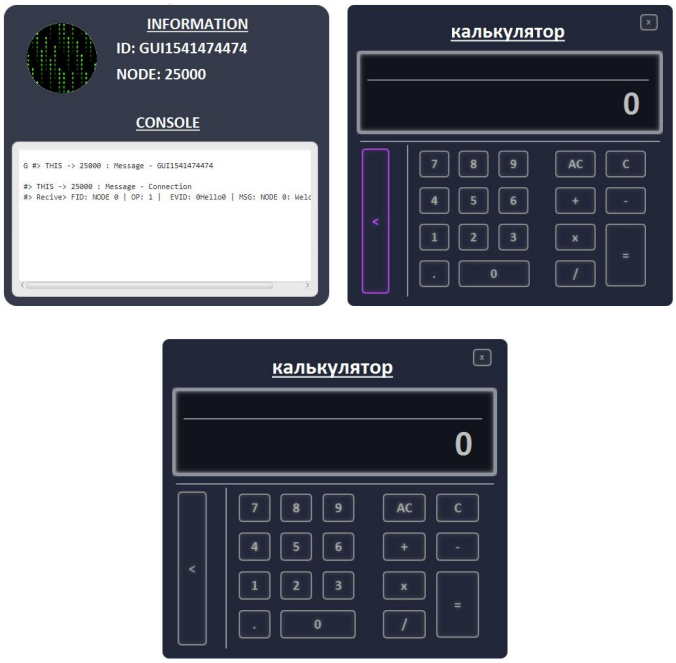


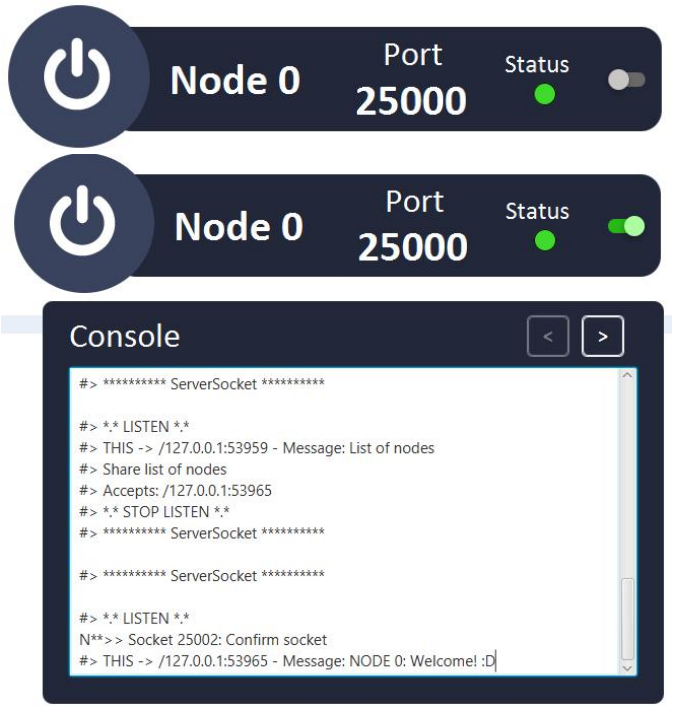
Fig. 13 Esquema del sistema

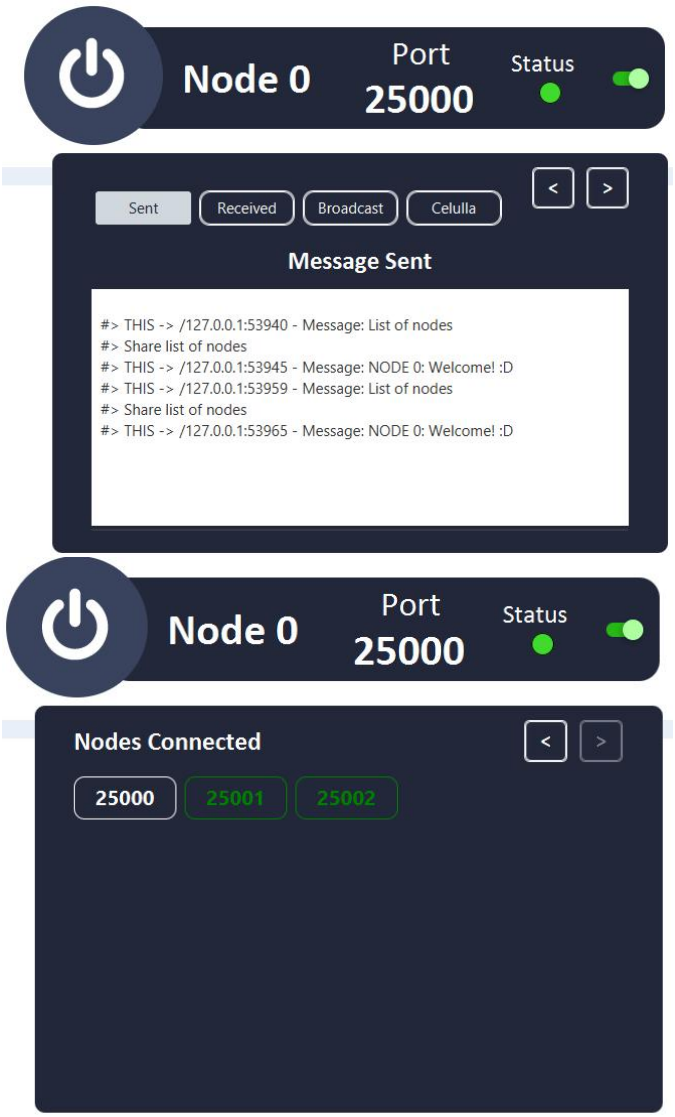
Gracias al conocimiento adquirido durante la implementación de este proyecto se han mejorado las interfaces gráficas y ya planearon las vistas de los nuevos modelos de cada uno de los componentes dando como resultado las siguientes vistas o interfaces.

Interfaz gráfica de la calculadora:

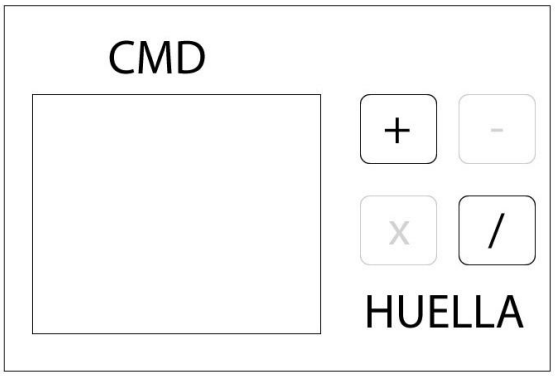


Interfaz gráfica nodos:





Interfaz gráfica célula servicios (Concepto):



Interfaz manejador de células:

IV. Conclusiones

La calculadora junto con los demás elementos nos permiten llegar a desarrollar los más parecido que ofrece ADSOA completo, al lograr realizar el campo de datos y las diferentes células junto con los protocolos para su implementación, nos perimte realizar una calculadora con sistema críticos.

Además se ha demostrado que un proyecto muy sencillo se puede lograr un sistema robusto y bien eleborado, en esta ocación ocupamos Java por su gran compatibilidad con los conceptos de esta arquitectura pero se puede llevar a otros lenguajes y aplicar en todas las circuntacias.

El proyecto no ha concluido, aún falta mucho por implementar y aumentar a este proyecto. Espero alguna vez poder retomarlo y poder contribuir en el futuro.