RIELAC
Revista de Ingeniería Electrónica, Automática y Comunicaciones

Propuesta de arquitectura para la capa de Computación al Borde en entornos de Industria 4.0

Zuliet Medina Rodríguez, Hernán David Pérez Villanueva, Alberto Sergio Prieto Moreno

RESUMEN / ***ABSTRACT***

En el marco de la Industria 4.0 se desea lograr que los procesos sean completamente automatizados. En otras palabras, que las empresas se encuentren en el cuarto nivel de la pirámide de automatización clásica. Con este fin, la empresa EMSI FARMA está desarrollando un sistema de gestión de operaciones de fabricación. Para el correcto funcionamiento del sistema se ha dividido en cuatro capas: nube, niebla, borde y dispositivos de campo. El siguiente artículo tiene como objetivo realizar una propuesta de arquitectura para la capa de Computación al Borde que cumpla las necesidades del sistema. La arquitectura de Borde que se propone se encarga de acercar la lógica de las operaciones a los dispositivos de campo. Para el desarrollo de la misma se utiliza un diseño de bloques funcionales, donde cada elemento tiene sus propias responsabilidades. Con el desarrollo de la arquitectura se logró abstraer la lógica de los elementos del protocolo de la comunicación que se utiliza para conectarse con las capas superiores e inferior. Además, se logra que el seguimiento de las operaciones y la adquisición de los datos se encuentre lo más cercano posible de la fuente de datos.

Palabras claves: Computación al Borde, Industria 4.0, Internet Industrial de las Cosas

Within the framework of Industry 4.0, it is desired to ensure that processes are completely automated. In other words, companies are at the fourth level of the classic automation pyramid. To this end, the EMSI FARMA company is developing a manufacturing operations management system. For the correct behavior of the system, it has been divided into four layers: cloud, fog, edge and field devices. The following article aims to make an architectural proposal for the Edge Computing layer that meets the needs of the system. The proposed Edge architecture is responsible for bringing the logic of operations closer to the field. For its development, a design of functional blocks is used, where each element has its own responsibilities. With the development of the architecture, it was possible to abstract the logic of the elements of the communication protocol that it uses to connect with the upper and lower layers. In addition, it is achieved that the monitoring of operations and the acquisition of data is as close as possible to the data source.

Keywords: Edge Computing, Industry 4.0, Industrial Internet of Things

Architecture proposal for the Edge Computing layer in Industry 4.0 environments

1. -INTRODUCCIÓN

El avance de las tecnologías, y el desarrollo de los procesos automatizados dieron paso a la cuarta revolución industrial, Industria 4.0 (I4.0). La I4.0 tiene como objetivo la automatización de los procesos de fabricación/producción, la creación de cadenas de valor digitales de extremo a extremo, y el mantenimiento de la línea de producción en tiempo real [1]. Lo que permite que el soporte tecnológico y la intervención en el equipamiento y los procesos se realice lo más inmediatamente posible [2]. Esto se logra gracias a la integración de los diferentes niveles de la pirámide clásica de automatización con los conceptos de Internet Industrial de las Cosas (IIoT, por sus siglas en inglés) [3].

Recibido: 03/2024 Aceptado: 07/2024

El IIoT se puede definir como un sistema que comprende objetos inteligentes en red, activos ciber-físicos, tecnologías de información genéricas que se asocian, y plataformas informáticas en la nube o en el borde. Estos permiten el acceso, la recopilación, el análisis inteligente y autónomo de datos, y las comunicaciones en tiempo real. Así como el intercambio de información de procesos, productos o servicios, dentro del entorno industrial, a fin de optimizar el valor total de la producción [4]. Esta comunicación en tiempo real debe ser vista en relación con la dinámica de los procesos (si son de dinámica rápida o lenta) y el tiempo de muestreo que debe utilizarse en cada caso. De esta forma se logra realizar un uso eficiente de los sistemas de almacenamiento de datos. Lo que es demandado en la Industria 4.0 para soluciones de monitoreo de condición y ciberseguridad entre otras aplicaciones [5-6].

Con los avances científicos-tecnológicos del mundo y la necesidad de aumentar la producción de la industria nacional es necesario desarrollar un sistema de gestión de operaciones de fabricación (MES, por sus siglas en inglés). Este sistema se ubica en la cuarta capa de la pirámide de automatización. Como su nombre lo indica, el sistema MES, tiene por objetivo realizar de forma automática la gestión de las operaciones de la industria donde se despliegue. La empresa EMSI FARMA se dio a la tarea de desarrollar un sistema MES para el ámbito de I4.0, donde se tienen en cuenta los requerimientos de la industria farmacéutica [7].

Entre los principales requerimientos se encuentran la digitalización de datos. Lo que puede incluir información relacionada con: la cadena de suministro, la variabilidad de las materias primas y el seguimiento global de los materiales en todas las instalaciones, la información relacionada con las condiciones ambientales de fabricación, los procedimientos operativos, y las instrucciones de trabajo del operador. Además, monitorear operaciones en tiempo real y centralizar datos de eventos de auditoría, para mejorar la toma de decisiones y un aseguramiento continuo de la calidad del proceso. Esto permite reducir la variabilidad entre lotes, producir productos disponibles de manera consistente y una mayor flexibilidad y agilidad de producción. La arquitectura del sistema que se desarrolla teniendo en cuenta estos requerimientos se divide en 4 capas como muestra la Figura 1.

Figura 1

Arquitectura del Sistema de gestión de operaciones de fabricación.

- Nube: Realiza, desde Internet, las configuraciones de los procesos y la visualización de la información.
- Niebla: Almacena los datos históricos y proporciona un seguimiento a los procesos, funciona como intermediario entre Internet y la capa de Borde.
- Borde: Acerca el procesamiento y la toma de decisiones a los dispositivos de campo.
- Dispositivos de campo: Se conforman por las diferentes fuentes de datos, sensores, actuadores y otros dispositivos físicos automatizados que se encuentran en la industria.

La capa de Computación al Borde tiene gran importancia ya que permite realizar el seguimiento de las operaciones, aunque existan fallas de comunicación con los servicios de la nube. Acerca la toma de decisiones y el procesamiento de la información a los dispositivos de campo. Disminuye el tráfico de información hacia la nube [8]. Además, teniendo en cuenta la arquitectura que se propone, esta capa no se encuentra conectada directamente a Internet lo que reduce las posibilidades de ataques a los procesos de producción y la fuga de información. Entre los principales requerimientos de usuario a tener en cuenta, para el desarrollo de la capa de Computación al Borde del sistema de gestión de operaciones de fabricación que está en desarrollo, se encuentran:

- Comunicación entre sus elementos internos.
- Comunicación con diferentes protocolos industriales.
- Gestión de fuentes de datos que sufrieron desconexión.
- Gestión de operaciones con diferentes niveles de complejidad (operaciones de unidad, procedimientos de unidad y procedimientos de centros de trabajo).
- Presencia de un sistema de adquisición de datos y de gestión de eventos de auditoría.
- Persistencia de la información.

El principal aporte del siguiente artículo es la propuesta de arquitectura para la capa de Computación al Borde para un sistema MES, que satisface los requerimientos fundamentales para la industria farmacéutica mencionados anteriormente. Además, se definen las funciones de cada una de las partes de dicha arquitectura.

La estructura del trabajo es la siguiente: en la sección 2 se presenta un análisis de antecedentes donde se realiza un estudio de arquitecturas consultadas y los requerimientos que satisfacen. En la sección 3 se presenta la propuesta de arquitectura de Computación al Borde donde se explica la ubicación física de la capa de Borde, la integración entre los elementos que la conforman y la tecnología utilizada para su implementación. Finalmente se presentan las conclusiones.

2.- ANÁLISIS DE ANTECEDENTES SOBRE LA CAPA DE COMPUTACIÓN AL BORDE

En los últimos años ha existido un aumento del tráfico de información hacia la nube, el cual continuará creciendo en los años venideros. Además, algunas aplicaciones comenzaron a producir gran cantidad de datos que necesita ser almacenada y surge la necesidad de tiempos de respuesta cortos y su manejo de manera privada. Al no ser la computación en la nube lo suficientemente eficiente para soportar estos cambios, nace el concepto de Computación al Borde [9-10].

Según [10] se define "borde" como cualquier recurso informático y de red a lo largo del camino entre las fuentes de datos y los centros de datos en la nube. Visto de esta forma se pudiera pensar que se habla de la Computación de Niebla. La diferencia radica en que la Niebla se enfoca en el lado de la infraestructura, mientras la Computación al Borde lo hace hacia el lado de los dispositivos de campo [11-12].

El enfoque de Borde reduce efectivamente la latencia y el tráfico de red y acerca el procesamiento y almacenamiento de datos a la fuente que los genera [13-14]. Esta proximidad facilita la toma de decisiones en tiempo real y mejora la confiabilidad del sistema. Para el diseño de la propuesta de la arquitectura de Borde, que se explicará en este artículo, fue necesario realizar una investigación del conjunto de requerimientos que satisfacen arquitecturas utilizadas a nivel mundial. De esta forma conocer si alguna cumple las necesidades del sistema MES a desarrollar. A continuación, se presentan las arquitecturas que se consultaron.

2.1.- ARQUITECTURAS DE REFERENCIA

FAR-Edge RA (Figura 2) es un marco conceptual para el diseño e implementación de la plataforma del proyecto FAR-Edge que se basa en la Computación al Borde. Entre sus conceptos principales se encuentran alcances y niveles. Por un lado, los alcances se refieren a componentes de una planta o sus ecosistemas como maquinaria, dispositivos de campo, estaciones de trabajo [15-16]. Por otro lado, los niveles proporcionan información sobre las partes individuales del sistema. Como limitante, no se muestra la correlación entre cada uno de los elementos de las capas. Por tal motivo, no queda claro cómo se entrelazan para conformar la capa de Borde.

Figura 2

Arquitectura FAR-Edge RA [16].

Edge Computing RA 2.0 es una arquitectura en la que el enfoque vertical se dirige a los servicios de gestión, ciclo de vida de datos y seguridad, que se orientan a servicios inteligentes a lo largo de todo el proceso. Sigue un modelo horizontal de capas con interfaces abiertas como se ve en la Figura 3. Se basa en el diseño de arquitecturas a través de bloques funcionales, donde cada bloque se caracteriza por sus entradas, salidas y funciones internas lo que permite tener las responsabilidades separadas [15-16]. Edge Computing RA 2.0 muestra la relación entre los elementos de las capas. Por otra parte, no se refleja la comunicación entre la capa de Borde con las fuentes de datos que es un aspecto imprescindible. El proyecto en desarrollo necesita tener la capacidad de comunicarse con fuentes de datos que presentan diferentes protocolos de comunicación. Esto se debe a que en entornos industriales el equipamiento existente puede ser de diferentes fabricantes que no utilicen los mismos protocolos.

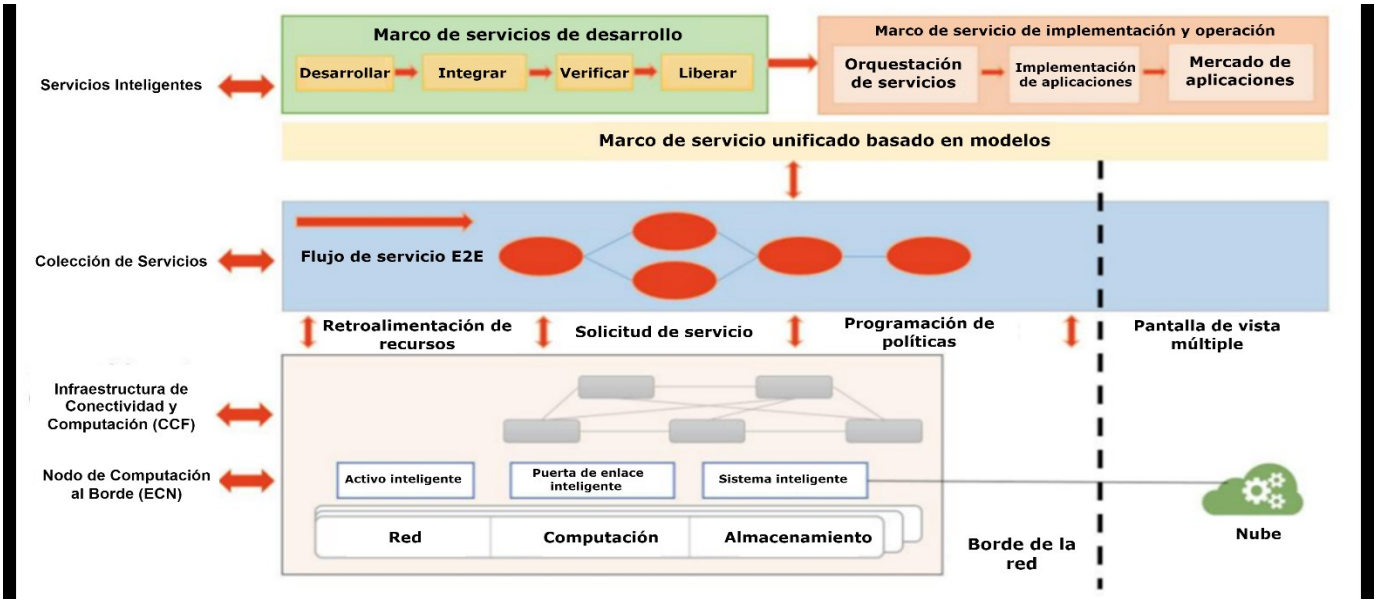

Figura 3

Arquitectura Edge Computing RA 2.0. Modificado de [15].

La arquitectura de referencia de **Edge Computing 3.0** presenta el contenido desde diferentes perspectivas de forma multi-vista (Figura 4). La funcionalidad de cada capa se muestra a través de la perspectiva funcional multicapa. La capa de Borde consta de dos componentes principales: el nodo de borde y el administrador de borde. El nodo perimetral representa el hardware de la entidad y se encarga de ejecutar las operaciones de computación perimetral. Por otro lado, el gestor perimetral utiliza principalmente software para gestionar de manera uniforme los nodos. Los nodos de Edge Computing 3.0 se dividen según las características del hardware y los tipos de servicio que ofrecen. Además, se implementan llamadas de funcionalidad genérica mediante interfaces de programación de aplicaciones (API). El módulo de funciones de dominio de control, análisis y optimización se utiliza para transmitir información entre las capas superior e inferior y para planificar los recursos de borde local [17]. De igual forma que en la arquitectura anterior no se refleja si es posible la abstracción de la comunicación con el tipo de protocolo que empleen las fuentes de datos. Un requerimiento de usuario para entornos de industria farmacéutica es la necesidad de darle atención a las fuentes de datos que sufrieron desconexión. Por ejemplo, es necesario comenzar un proceso de reconexión y si se están realizando operaciones en la fuente de datos desconectada es necesario conocer el estado de la misma cuando se restablece la conexión. En la arquitectura Edge Computing 3.0 no se menciona como se lleva a cabo la gestión de las fuentes de datos que fueron desconectadas.

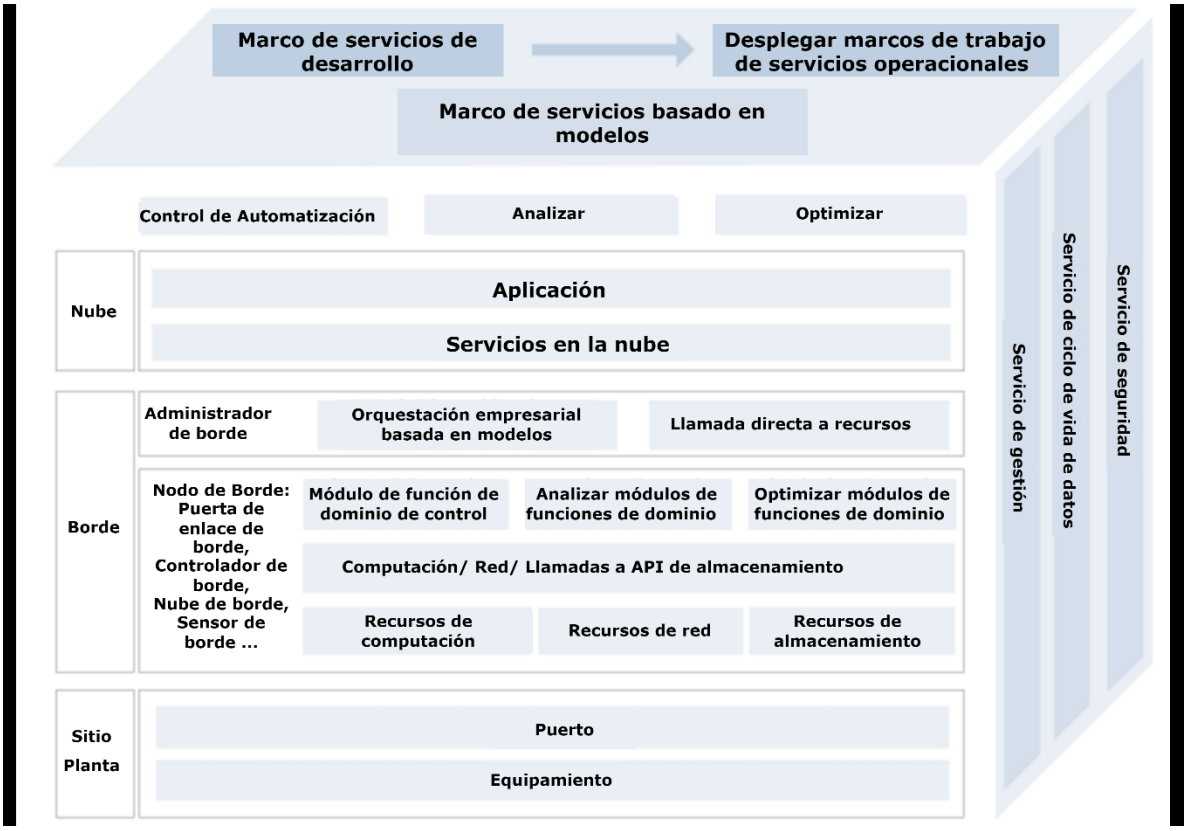

Figura 4

Arquitectura Edge Computing 3.0. Modificado de [17].

2.2.- PLATAFORMA DE CÓDIGO ABIERTO

Además del estudio de arquitecturas de referencia, se consideró la utilización de una plataforma de código abierto que permitiera ahorrar tiempo a la hora de realizar la implementación de la solución**. EdgeX Foundry** es un proyecto neutral que se aloja en la fundación Linux y es un marco abierto universal para la informática en el Borde de Internet de las cosas. Se ubica en una plataforma de software de referencia que es completamente independiente del hardware y los sistemas operativos (deficiencia que se encontró en las otras arquitecturas).

EdgeX Foundry consta de una colección de microservicios que se dividen en cuatro capas de servicios y dos servicios de sistema (Figura 5). Desde una perspectiva vertical, el kit de desarrollo de software (SDK) que la capa de servicio del dispositivo proporciona se utiliza para establecer los enlaces de comunicación con el "Southbound". La capa de servicio del dispositivo convierte los datos del dispositivo y los envía a la capa de servicio central. También puede recibir comandos de otros microservicios y pasarlos a dispositivos [18]. Incluye múltiples métodos de acceso, como el Protocolo de transmisión de telemetría de cola de mensajes (MQTT), el Dispositivo virtual (VIRTUAL) y Bluetooth de baja energía (BLE).

Se pudiera decir que es el mecanismo correcto para la solución de Industria 4.0 que se desarrolla por las ventajas que presenta. El problema radica en que la arquitectura de EdgeX cuenta con una implementación flexible que es una desventaja para proyectos muy específicos y complejos como el que se desarrolla. Esto ocasiona que el trabajo del programador se ralentice y sea necesario agregar a esta arquitectura nuevas funcionalidades para cumplir todos los requerimientos de usuario. Además, todos sus microservicios están muy integrados entre sí y poseen secciones donde la comunicación se realiza por medio de mensajes de bus que utilizan el protocolo MQTT. En la solución propuesta no se utiliza este protocolo por lo que se complejiza la acción de tomar uno o varios microservicios e integrarlos con un diseño al cual haya que agregarle capas.

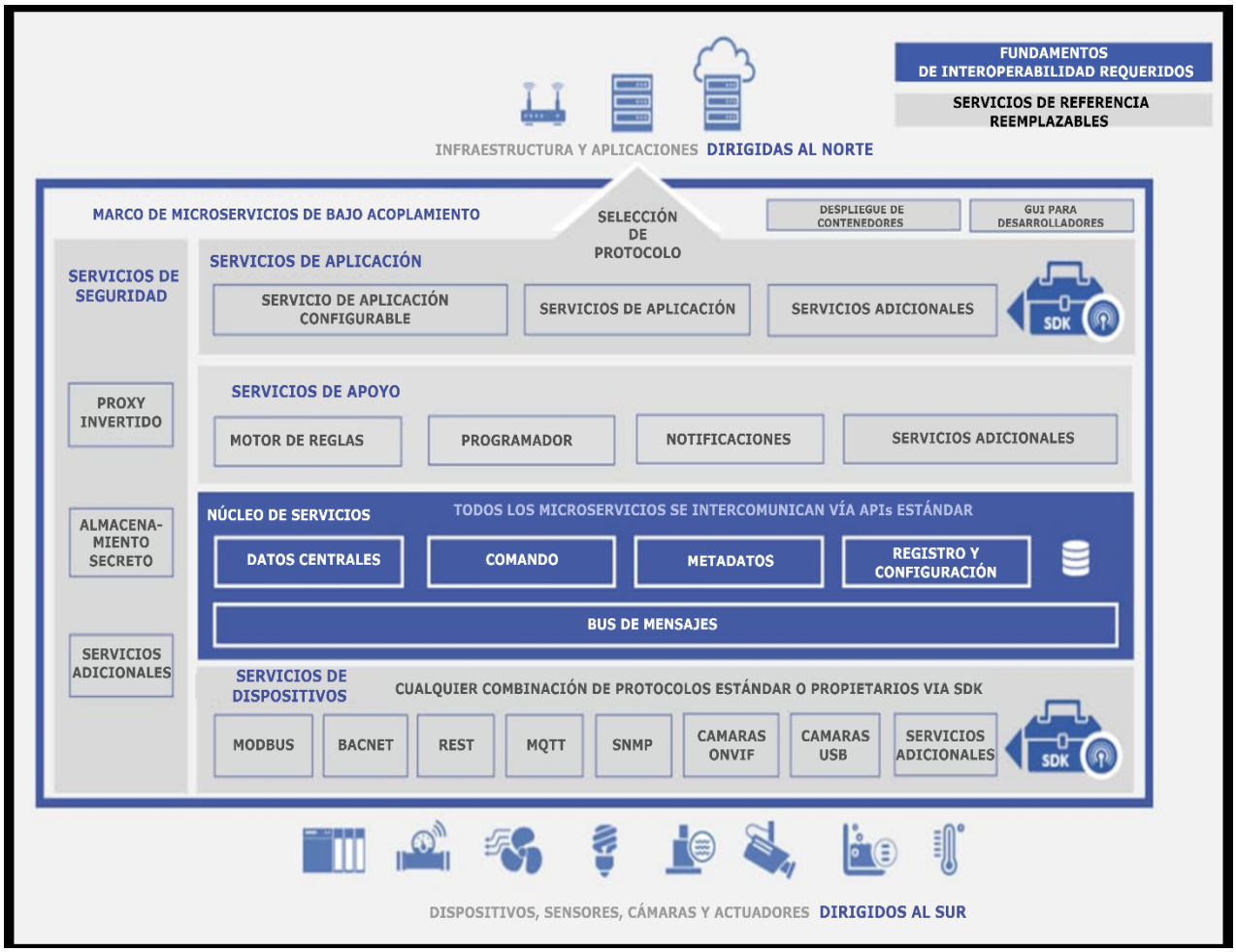

Figura 5

Arquitectura de la plataforma EdgeX Foundry. Modificado de [18].

La Tabla 1 muestra un resumen de cómo las arquitecturas consultadas cumplen o no con los principales requerimientos de usuario que presenta la solución de IIoT.

Las arquitecturas que se consultaron no compensan todos los requerimientos de usuario de la solución de IIoT que está en desarrollo como se resume en la Tabla 1. Por tal motivo, se propone una arquitectura que satisfaga las necesidades del sistema. En la propuesta se tienen en cuenta las fortalezas de las soluciones que se analizaron.

Tabla 1
Cumplimiento de los requisitos de usuario por parte de las arquitecturas estudiadas

Requerimientos de Usuario	FAR-Edge RA.	Edge Computing RA 2.0	Edge Computing 3.0	EdgeX Foundry
Comunicación entre sus elementos internos	No	Sí	No	Sí
Comunicación con diferentes protocolos industriales	No	No	No	Sí
Gestión de fuentes de datos que sufrieron desconexión	No	No	No	No
Gestión de operaciones con diferentes niveles de complejidad	No	No	No	No
Sistema de adquisición de datos	Sí	Sí	Sí	Sí
Sistema de gestión de eventos de auditoría	No	No	No	No
Persistencia de la información	Sí	No	No	Sí

3.- PROPUESTA DE LA ARQUITECTURA DE COMPUTACIÓN AL BORDE

A continuación, se describe la arquitectura de Computación al Borde que se propone. Para esta se tienen en cuentas los requerimientos de la solución de Industria 4.0 que se desarrolla. Se utiliza el enfoque de bloques funcionales, lo que permite realizar una separación de las responsabilidades, siendo cada bloque el encargado de una tarea específica. De esta forma es posible ampliar la solución al agregar nuevas funcionalidades sin afectar las existentes.

3.1.- UBICACIÓN FÍSICA DE LA CAPA DE BORDE

Es necesario conocer la estructura física de la empresa para entender la distribución de la capa de Borde. Según la norma ISA 95.00.03 del año 2005 [19] una empresa puede estar formada por sitios. A su vez estos sitios se dividen en áreas que contienen centros de trabajo. Cada centro de trabajo tiene que tener al menos una unidad de trabajo como se muestra en la Figura 6. Además, en la Figura 7 tomada de la norma ISA 95.00.01 del año 2000 [20] se muestra que estas unidades de trabajo son las que se encuentran conformadas por equipos de niveles inferiores (fuentes de datos) utilizados en la operación.

Al analizar la estructura física de una empresa se aprecia que las fuentes de datos pueden encontrarse separadas por distancias pequeñas (en la misma unidad de trabajo) o distribuidas en diferentes áreas. Al tener en cuenta esta distribución se decide que la capa de Borde va a estar conformada por Módulos de Borde (*IIoTModule*). Un *IIoTModule* es un servicio informático que se despliega en cada una de las áreas de la empresa. De esta forma se cumple el principio de que la capa de Borde es la que se encarga de acercar el procesamiento de la información a la fuente de datos. La distribución de las capas de la solución de I4.0 queda como se muestra en la Figura 8. En esta se aprecia que la comunicación entre los *IIoTModule* y la niebla es a través

del mecanismo nombrado *IIoTHub*. El *IIoTHub* se encarga de decodificar los mensajes y enviarlo hacia los elementos restantes de la niebla. Además, es el responsable de codificar los mensajes provenientes tanto de los otros elementos de la niebla como de la nube y enviarlo al *IIoTModule* correspondiente.

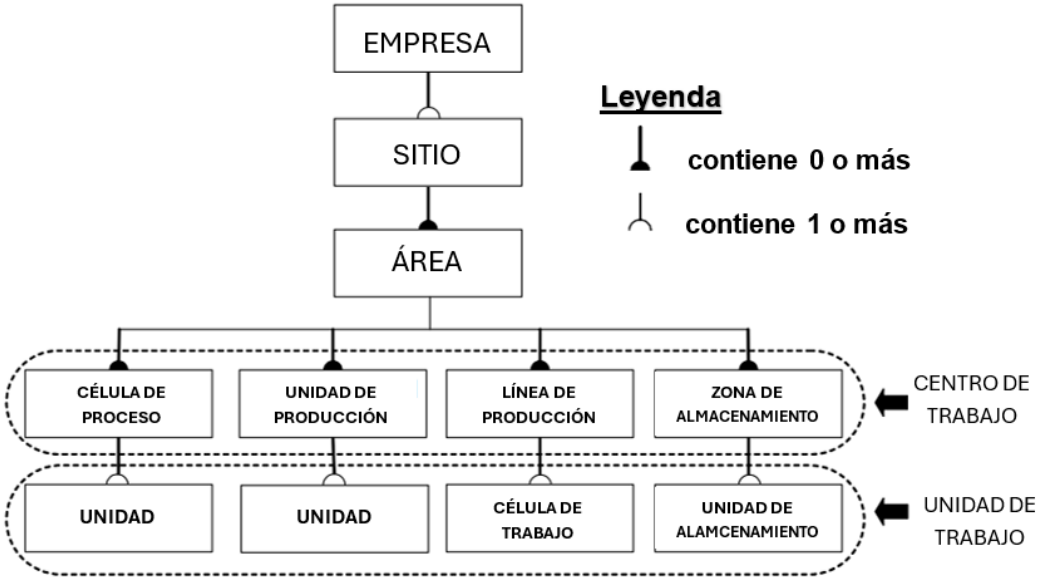

Figura 6

Estructura física de una empresa (ISA 95.00.03). Modificado de [19].

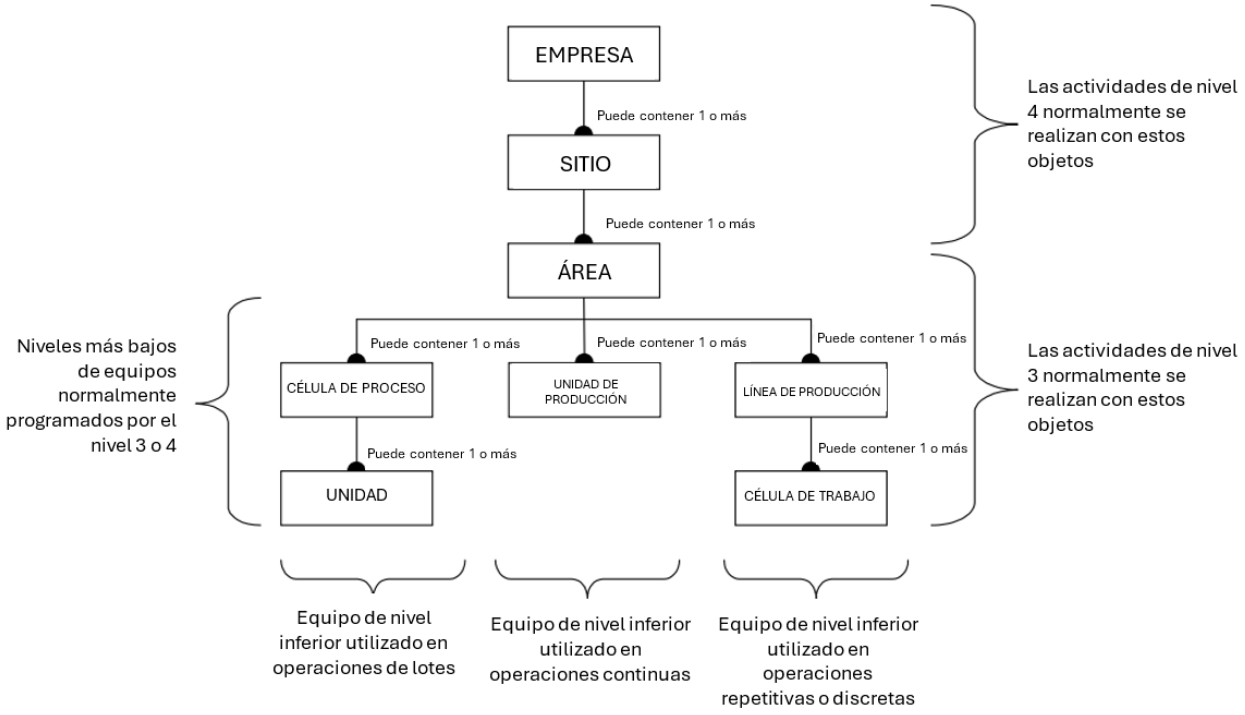

Figura 7

Estructura física de una empresa (ISA 95.00.01). Modificado de [20].

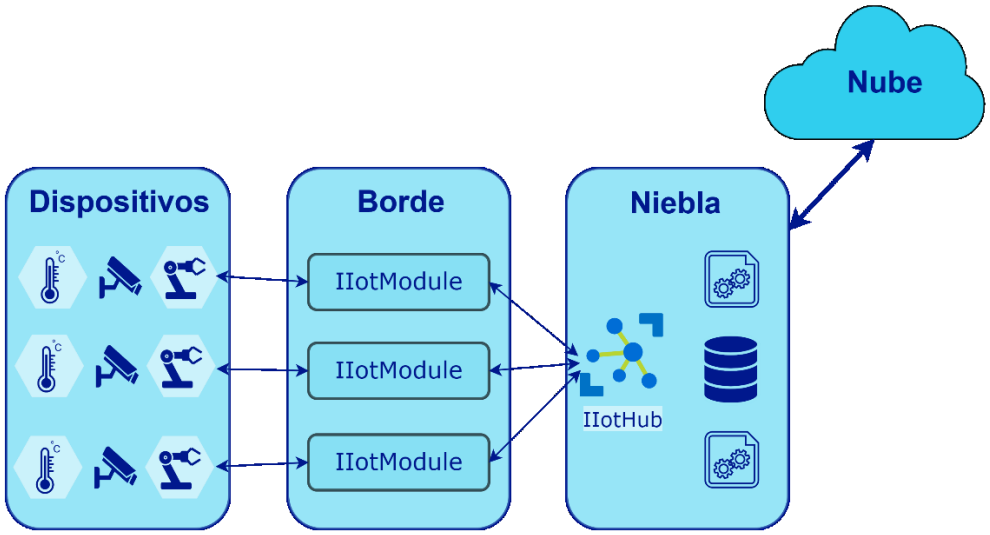

Figura 8

Capas de la solución de I4.0.

3.2.- ARQUITECTURA PARA UN MÓDULO DE BORDE

La arquitectura que se propone para los Módulos de Borde se conforma por bloques funcionales. Cada uno de estos bloques es el encargado de realizar una tarea específica. Esto permite tener las responsabilidades separadas y una mayor flexibilidad a la hora de ampliar la solución. En la Figura 9 se muestra la arquitectura a desarrollar para un *IIoTModule* ubicado en la capa de Borde.

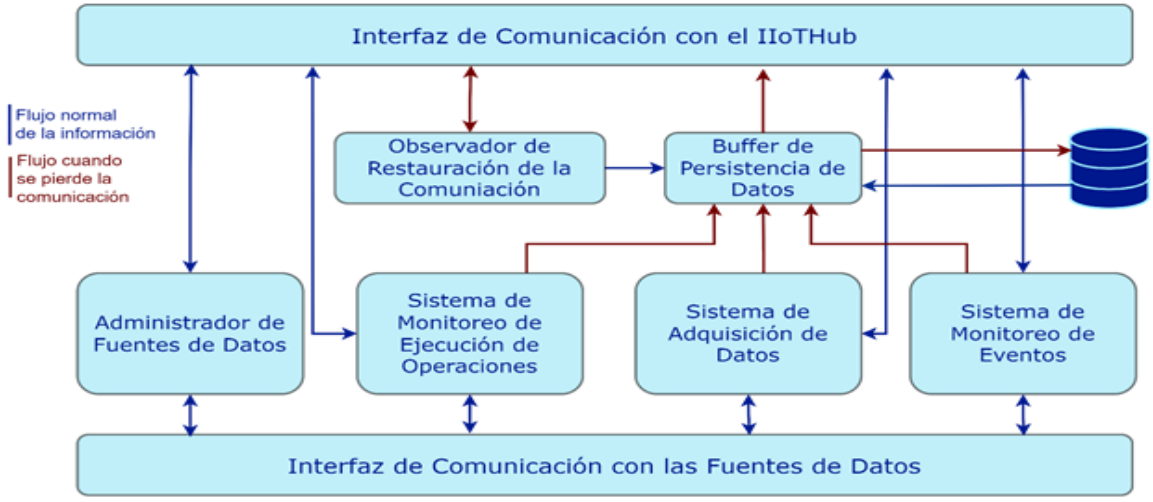

Figura 9

Propuesta de arquitectura de un módulo de Borde.

La comunicación entre los elementos internos del *IIoTModule* se realiza a partir de dos mecanismos, uno de solicitud/respuesta y otro de notificación. El mecanismo de notificación además de emplearse cuando el remitente no necesita una respuesta, también se utiliza para aquellas acciones que deban ser atendidas por varios bloques de la solución. Cada bloque le da una

atención a la notificación acorde a sus funcionalidades. A continuación, se describen las funcionalidades de cada uno de los bloques mostrados en la Figura 9.

Interfaz de Comunicación con el IIotHub (*IIoTHubInterface***)**: Permite realizar la abstracción del protocolo de comunicación que se utiliza para comunicar el servicio del *IIoTModule* con la niebla. Esta se encarga de codificar los mensajes que deben ser enviados hacia el *IIoTHub*. Los mensajes se nombran Eventos de Integración y permiten enlazar elementos de capas diferentes. Entre los Eventos de Integración se encuentran los que contienen los valores de las muestras, el cambio de estado de las operaciones, el cambio de estado de las fases, el cambio de modo de las operaciones, la detección de eventos de auditoría, la detección de alarmas y el cambio de estado de una alarma. Además, el *IIoTHubInterface* también se encarga de decodificar la información proveniente de las capas superiores. Entre los Eventos de Integración que debe decodificar se encuentran la adición, actualización y eliminación de fuentes de datos correspondientes al *IIoTModule*; y la habilitación de una operación en una fuente de datos específica. Una vez decodificados los Eventos de Integración, el *IIoTHubInterface* informa a los demás bloques de su ocurrencia. De esta forma si es necesario cambiar el protocolo de comunicación no se afectan los demás mecanismos ni la lógica utilizada en el *IIoTModule*.

Administrador de Fuentes de Datos: Como su nombre lo indica es el responsable de manejar las fuentes de datos. Además, da respuesta a las solicitudes de adicionar fuentes de datos, eliminarlas o actualizarlas. Cuenta con 2 mecanismos, uno para las fuentes de datos que son accesibles (aquellas con las cuales logró establecer comunicación) y otro para las fuentes de datos inaccesibles. Cuando se realiza la configuración de una nueva fuente de datos pasa a ser inaccesible hasta que se logra establecer comunicación. También notifica al operario, a partir del indicador visual de la fuente de datos, que se lleva a cabo una comunicación exitosa. En caso de que se pierda la comunicación es el manejador el responsable de descubrir que se reestableció mediante un proceso de ping.

Sistema de Monitoreo de Ejecución de Operaciones (OEMS, por sus siglas en inglés): Administra los mecanismos de monitoreo de ejecución de operaciones. Al manejar la notificación de operación habilitada, que el *IIoTHubInterface* lanza, el OEMS crea un Monitor de Ejecución de la Operación (OEM, por sus siglas en inglés) (Figura 10). El OEM se encarga de monitorear una operación en específico, o sea, detectar los cambios de estado de las operaciones y las fases y los cambios de modo de las operaciones. Además, traduce la posible receta que se debe ejecutar en el autómata. Para los procesos automatizados se considera receta a un conjunto de información necesaria que define de forma única los requisitos de producción para un producto específico.

Para realizar la traducción de la receta se cuenta con un grafo que organiza y convierte, a lenguaje conocido por el autómata, el orden de las fases a ejecutar y los parámetros de estas. Un parámetro es una de las variables que interviene en el proceso que se ejecuta. La receta puede tener tres niveles de complejidad:

1. Simple: Solamente se realiza un seguimiento de las fases de la operación ya que se encuentran predefinidas en la fuente de datos. Por tal motivo el OEM no cuenta con una receta que traducir.
2. Normal: Todas las fases que se ejecutan se hacen de forma lineal, como si fuera una cola, la primera que entra es la primera que sale.
3. Complejo: Se pueden ejecutar fases en paralelo (dos o más a la vez) o pueden existir condiciones externas por las que es necesario esperar antes de llevar a cabo la siguiente fase. Un ejemplo es esperar por el consentimiento del operario.

En el contenido de una receta se encuentran los niveles superiores e inferiores que pueden tomar los parámetros. Esto se debe a que para la fabricación de un producto en fases diferentes, que no se ejecuten en paralelo, una misma variable puede tener disímiles intervalos de funcionamiento. Por otra parte, el OEMS es el responsable de manejar los cambios de estado de las fases de las operaciones, que están en ejecución, y por ende de las operaciones. Además de manejar estos cambios le notifica al *IIoTHubInterface* para que los transmita a las capas superiores. Si el estado de la operación es abortado o completado se elimina al OEM correspondiente.

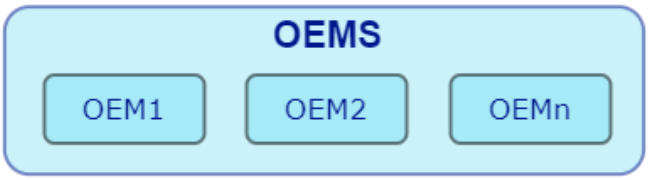

Figura 10

Estructura del OEMS.

Sistema de Adquisición de Datos (DAS, por sus siglas en inglés) Es el responsable de la adquisición de las muestras de las variables. Para esto define Mecanismos de Adquisición de Datos (DAM, por sus siglas en inglés) por cada operación que se inicia (Figura 11). El DAM se encarga de manejar el tiempo con el que deben ser medidas las variables de una operación. Cada fuente de datos contiene una base de muestreo, que, unida al multiplicador de la operación y al multiplicador propio de la variable, establece el período de muestreo de esta. La variable incorpora información al muestreo debido a si son de dinámica más lenta o más rápida. De esta forma se evita el uso inadecuado de las fuentes de datos al medir variables de dinámica lentas como la temperatura y la humedad y variables de dinámica rápida como la presión y el caudal con la misma frecuencia.

Existen operaciones de distintas naturalezas, por lo que sus períodos de muestreo son diferentes. Las operaciones de seguimiento, donde se debe cumplir un perfil de comportamiento específico, necesitan una mayor frecuencia de muestreo. Mientras que en las operaciones de regulación es imprescindible una menor frecuencia de muestreo para no perder información de las variables. Por ejemplo, en un lazo de circulación de agua los muestreos son más esporádicos debido a que es una operación de seguimiento. Por otra parte, en una operación de sanitización del lazo es necesario que la frecuencia de muestreo sea menor ya que este proceso se realiza en un intervalo de tiempo menor.

Una vez trascurrido el período de monitoreo de una variable, el DAM notifica al DAS que contiene variables listas para la obtención de sus muestras y se reinicia el contador de la variable. En caso de que no existan notificaciones pendientes a ser solicitadas a las fuentes de datos, el DAS espera 30 segundos por otras notificaciones ante de realizar la petición de las muestras. De esta manera se agrupan varias solicitudes de muestreo. Una vez obtenidas las muestras se agrupan en un Evento de Integración de muestras obtenidas y se envían a las capas superiores. El DAS también maneja la notificación de cambio de estado de la operación. Si es pausada se pausa el temporizador del DAM correspondiente. Si es finalizada o abortada se elimina el DAM. En caso de que una fuente de datos salga de accesibilidad se pausan los DAM relacionados con las operaciones que se ejecutan en ella.

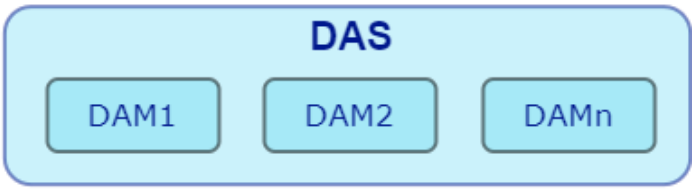

Figura 11

Estructura del DAS.

Sistema de Monitoreo de Eventos (EMS): Además de conocer el cambio de valor de las variables el sistema es responsable de atender la ocurrencia de eventos de auditoría en las fuentes de datos. El EMS es el mecanismo que se encarga de darle seguimiento a la ocurrencia de estos eventos y de las alarmas que son detectadas por la fuente de datos. Cuando una fuente de datos pasa al estado de accesibilidad se comienza con el mecanismo de monitorear los eventos de auditoría y las alarmas que puedan ocurrir. Entre los eventos de auditoría se consideran la autenticación de un usuario y el cambio de valor en alguno de los parámetros. Una alarma ocurre, en la mayoría de las ocasiones, cuando alguno de los parámetros de la operación sobrepasa los valores superiores o no alcanzan los valores inferiores configurados en la receta. Además, es necesario darle un seguimiento al estado de la alarma, de esa forma se conoce el cambio de estado de la misma. Entre los estados de una alarma se encuentran: activa, reconocida y recuperada. La información recopilada se convierte en Eventos de Integración y se envía a las capas superiores para su análisis y visualización.

Observador de Restauración de la Comunicación: Elemento que se encarga de identificar cuándo se ha recuperado la comunicación con la niebla una vez que esta se pierde. Maneja la notificación de comunicación fallida haciendo un proceso de encuesta al *IIoTHub* hasta que la comunicación se reestablezca. Una vez reestablecida le notifica al resto de los elementos, de esta forma es posible comenzar el proceso de envío de la información almacenada en el Buffer de Persistencia de Datos hacia las capas superiores.

Buffer de Persistencia de Datos: En caso de que la comunicación con la capa superior falle es necesario un mecanismo que se encargue de almacenar la información temporal referida a las ejecuciones de las operaciones, el muestreo de las variables y la ocurrencia de eventos o alarmas. En otras palabras, es necesario almacenar todos aquellos Eventos de Integración que en condiciones normales son enviados a las capas superiores. De esta forma no se pierde su contenido (Fig. 12). Cuando se reestablece la comunicación el Buffer es el responsable del envío de la información que tiene almacenada hacia las capas superiores de la solución de I4.0. Si el envío se realiza de forma satisfactoria la información se elimina de la base de datos temporal.

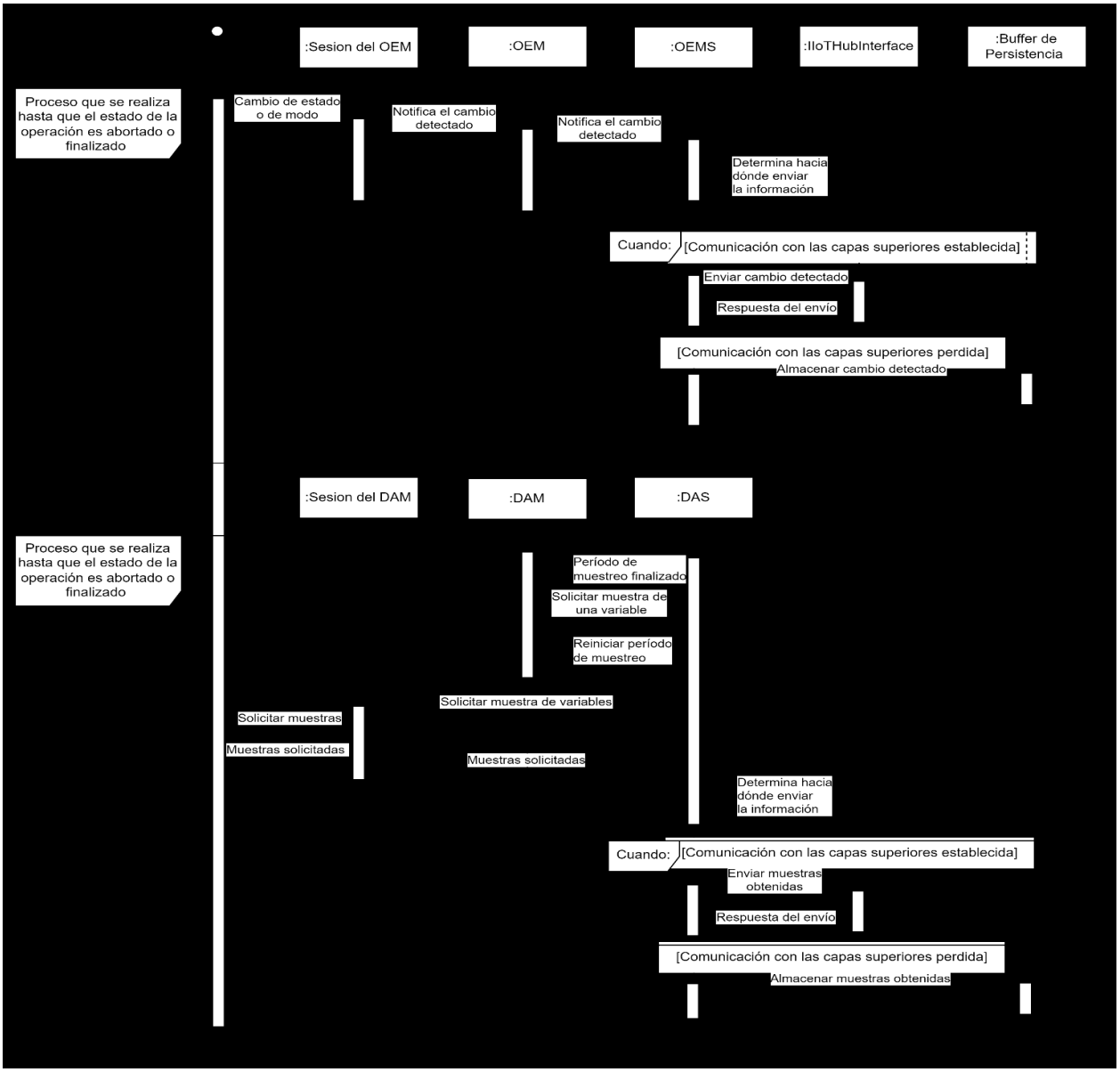

Figura 12

Diagrama de flujo de persistencia de los datos.

Interfaz de Comunicación con las Fuentes de Datos (*DataSourceCommunication*): Permite realizar la abstracción del protocolo de comunicación que tienen las fuentes de datos y así poder realizar acciones sobre la misma. Para establecer la comunicación se crea una Sesión de Comunicación con la Fuente de Datos. Una Sesión es el mecanismo que se encarga de traducir operaciones genéricas al correspondiente protocolo de la fuente de datos con la que desea interactuar. Soporta las operaciones de lectura y escritura desde y hacia una fuente de datos, así como la suscripción a cambios de valores. Cuando se recibe la notificación de operación habilitada el OMS y el DAS solicitan la creación de sesiones del tipo del protocolo de comunicación que soporta la fuente de datos. En el caso del EMS lo hace cuando se detecta una fuente de datos accesible. De esta forma, cuando se desee obtener, por ejemplo, el valor de una muestra, el DAS realiza la solicitud y la sesión correspondiente es la que se encarga de traducir esa solicitud a un lenguaje que conoce la fuente de datos. Entre los protocolos de comunicación industriales a los que se le da soporte se encuentran OPC UA, BACnet y Modbus (Fig. 13). Esto no quiere decir que son los únicos, ya que el mecanismo de Sesiones permite insertar nuevos protocolos sin cambiar la lógica del programa.

Figura 13

Estructura de la Interfaz de Comunicación con las Fuentes de Datos. Relación con otros componentes.

3.2.1.- INTEGRACIÓN DE LOS BLOQUES DEL *IIOTMODULE* ANTE UN INICIO DE OPERACIÓN

Como se ha mencionado hasta ahora la capa de Computación al Borde se encarga de acercar el control de las operaciones hacia las fuentes de datos. Por tal motivo se considera imprescindible explicar el flujo de la información ante la habilitación de una operación, en el cual intervienen todos los bloques del *IIoTModule*.

Cuando se habilita la operación se crea un OEM y un DAM para darle seguimiento a la operación y adquirir las muestras en el momento correspondiente, respetivamente. Para mantener la comunicación con la fuente de datos se crean Sesiones que conozcan el protocolo de comunicación de la misma. En caso de que la comunicación con las capas superiores falle se almacenan los Eventos de Integración en la base de datos temporal y cuando se reestablece la misma se envían el *IIoTHub*. En caso de que falle la comunicación con la fuente de dato, esta pasa a un estado de inaccesibilidad y se comienza el proceso de encuesta. Además, son pausados los DAM relacionados con esa fuente de datos. Cuando la operación finaliza o es abortada se eliminan el OEM y el DAM y las Sesiones de comunicación correspondientes. De esta forma se le da cumplimiento a todos los requerimientos de usuario que se analizaron en la Tabla 1.

En la Figura 4 se aprecia un diagrama de flujo para el proceso que se lleva a cabo ante la habilitación de una operación. Este es un diagrama ideal para los casos en los que no se existen fallos en la comunicación con los elementos superiores e inferiores. Se decide no agregar el EMS ya que las sesiones del mismo se crean por fuentes de datos y no por ejecución de operaciones.

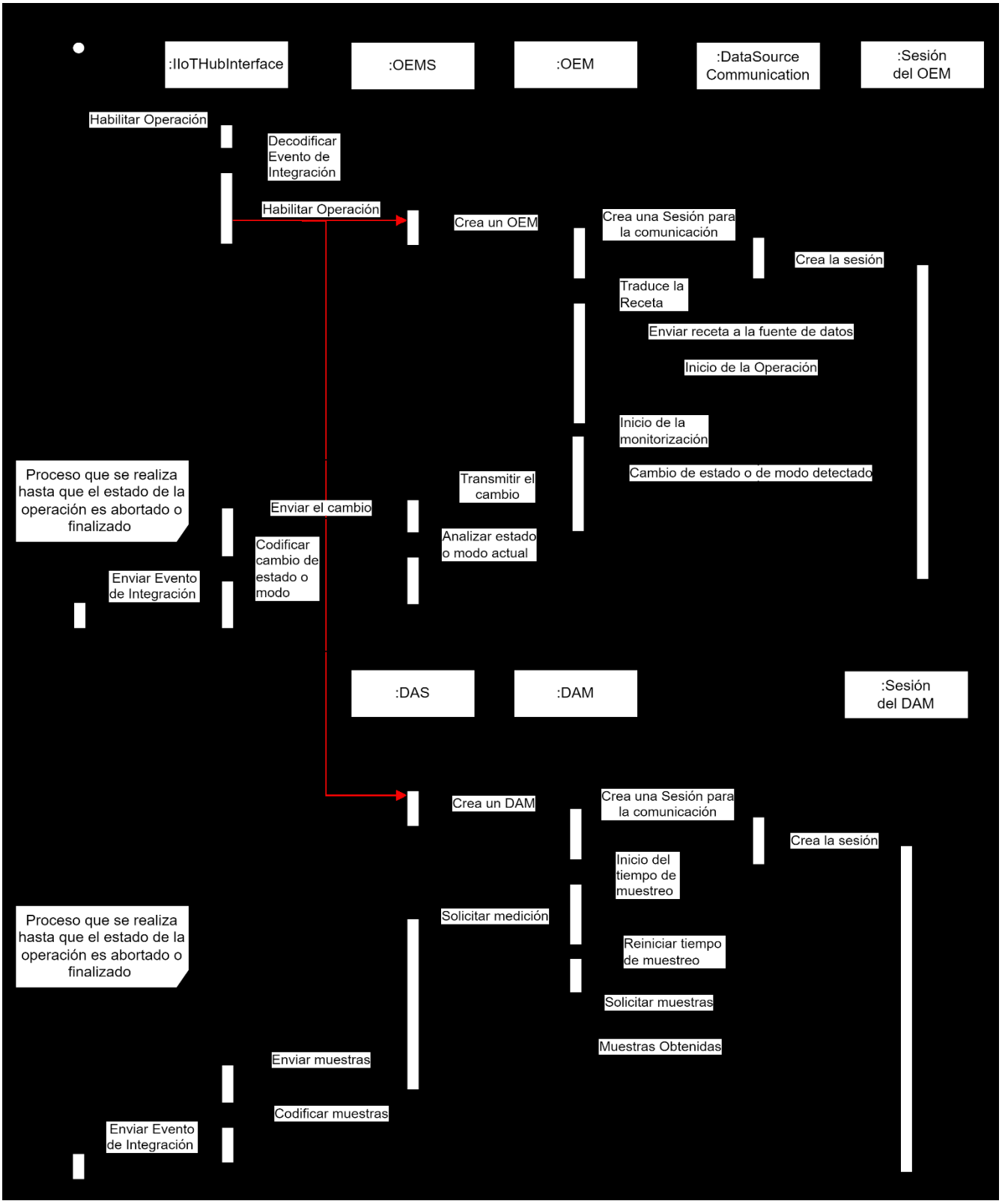

Figura 14

Diagrama de flujo de habilitación de una operación.

3.3.- SELECCIÓN DE LA TECNOLOGÍA E IMPLEMENTACIÓN DE LA ARQUITECTURA

La arquitectura propuesta de Computación al Borde se está implementando para el proyecto de I4.0 utilizando el lenguaje de programación C# y el *framework* .Net 8. Se sigue un diseño guiado por dominio (DDD, por sus siglas en inglés) y el uso de microservicios. Además, para la comunicación con el IIoTHub se emplea el protocolo gRPC y para la comunicación con las fuentes de datos, en una primera versión de la arquitectura, se implementan los protocolos OPC UA, BACnet y Modbus que son los más utilizados en la industria. En futuras versiones se incluirán otros protocolos.

El desarrollo basado en microservicios consiste en construir una aplicación con un conjunto de pequeños servicios que se ejecutan en su propio proceso (computación en hilos) y se comunican con mecanismos ligeros. Una gran ventaja del uso de microservicios es la idea de que cada uno contiene y maneja su propio estado [21]. De esta forma, cada bloque de un *IIoTModule* se comporta como un microservicio que al unirse conforman el servicio web del *IIoTModule* lo que permite seguir las tendencias actuales de tecnologías desacopladas como menciona [21].

Se escoge seguir un DDD ya que permite escribir código donde el dominio de la aplicación refleja los aspectos reales de la lógica de negocio. Esto facilita el mantenimiento del programa a largo plazo y satisfacer las necesidades futuras sin afectar los mecanismos actuales [22].

Como se menciona en [21, 23] gRPC es eficiente para la comunicación entre procesos ya que utiliza un protocolo binario basado en *protocol buffers* por lo que la transmisión de información es más rápida. Además, es fuertemente tipado, soporta múltiples lenguajes de programación y tiene soporte nativo para comunicación en directo tanto del cliente como del servidor, o de ambos a la vez. Según [21] es conveniente utilizar gRPC en proyectos donde intervengan la comunicación en tiempo real, los microservicios y dispositivos IoT, siendo estos elementos principales de la solución que se desarrolla. La comunicación utilizando el protocolo gRPC se realiza por canales de comunicación, donde cada cliente inicia la comunicación con el servidor y espera por una respuesta. En la Fig. 15 se aprecia como un servidor gRPC interactúa con sus clientes.

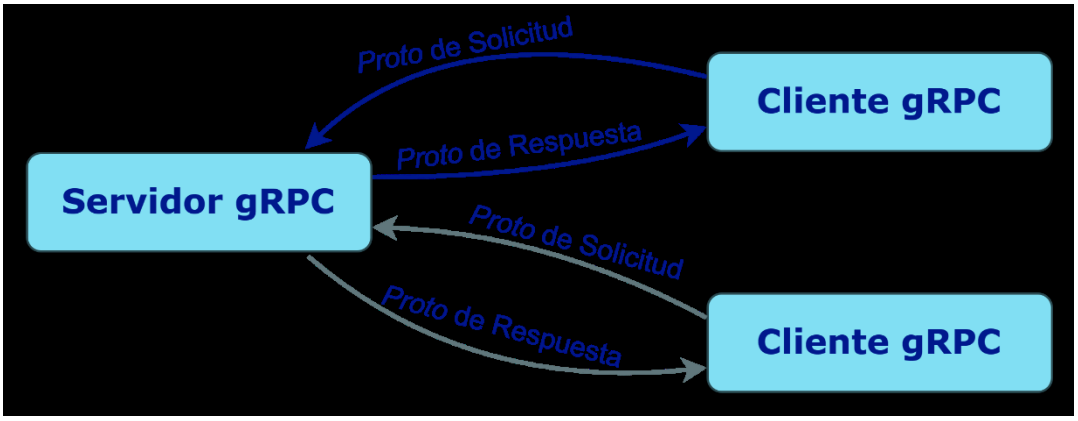

Figura 15

Comunicación cliente/servidor utilizando el protocolo gRPC.

Por otra parte, los autores deciden utilizar el *framework* .Net 8 por ser de código libre y por las amplias librerías y paquetes que contiene para el desarrollo de aplicaciones. Dentro de los paquetes que se utilizan para la implementación se encuentran:

- Grpc.Core y Grpc.AspNetCore: Para la comunicación con el *IIoTHub*,
- Quartz.Extensions.Hosting: para la programación multi-hilos de los diferentes servicios. Permite ejecutar tareas a partir de condiciones predefinidas.
- MediatR: Para la comunicación interna entre los bloques de la solución. Este permite lanzar una solicitud o notificación y tener diferentes manejadores que sean los encargados de darle respuesta.
- OpcFudation.NetStandar.Opc.Ua: para establecer comunicación con las fuentes de datos cuyo protocolo sea OPC
- BACnet: para establecer comunicación con las fuentes de datos cuyo protocolo sea BACnet
- Modbus.Net y NModbus: para establecer comunicación con las fuentes de datos cuyo protocolo sea Modbus

No se menciona la versión de los paquetes utilizados ya que a lo largo de la implementación puede variar por actualizaciones de los mismos.

4.- CONCLUSIONES

La arquitectura de Computación al Borde que se propuso para la solución de Industria 4.0 es la que se encarga de acercar la lógica de las operaciones al campo, para esto utiliza Módulos de Borde. En cada *IIoTModule* los elementos centrales de la arquitectura son capaces de abstraerse de los protocolos de comunicación dando paso a la utilización de diferentes protocolos en una misma solución. Con el diseño propuesto se logra cumplir no solo esta abstracción, sino que es capaz de configurar los parámetros de los dispositivos de campo para cada operación específica, al enviar hacia el autómata los valores del mismo. Cada *IIoTModule* le da seguimiento a diferentes operaciones, que se pueden encontrar ejecutándose en paralelo, de esta forma conocer el estado de sus fases, el modo en que se encuentra y su propio estado. Esto permite que el sistema responda y realice acciones ante los cambios detectados. Por otra parte, la distribución en bloques del recurso que se presenta permite la ampliación del mismo con nuevas funcionalidades en caso de ser necesario.

AGRADECIMIENTOS

Al equipo de trabajo de la empresa EMSI FARMA, en especial al departamento de Informática Industrial y a los miembros de la empresa AICA que forman parte del proyecto de Industria 4.0 que se lleva a cabo.

REFERENCIAS

1. Gupta S., Retracted: An edge-computing based Industrial Gateway for Industry 4.0 using ARM TrustZone technology. Journal of Industrial Information Integration. 2023, 33:1-5
2. Crespo, A.; Alonso, A. Una panorámica de los Sistemas de Tiempo Real. Revista Iberoamericana de Automática e Informática Industrial, 2006, 3(2): 7-18.
3. Becerra L.Y. Tecnologías de la información y las Comunicaciones en la era de la cuarta revolución industrial: Tendencias Tecnológicas y desafíos en la educación en Ingeniería. Entre Ciencia e Ingeniería. 2020; 14(28): 76-81
4. Boyes H., Hallaq B., Cunningham J., Watson T. The industrial internet of things (IIoT): An analysis framework. Computers in Industry. 2018, 101: 1-12
5. Quiñones-Grueiro M., Prieto-Moreno A., Llanes-Santiago O. Modeling and monitoring for transitions based on local kernel density estimation and process pattern construction. Industrial & Engineering Chemistry Research. 2016, 55(3): 692-702
6. Ares-Milián M.J., Quiñones-Grueiro M., Verde C., Llanes-Santiago O. A leak zone location approach in water distribution networks combining data-driven and model-based methods. Water.2021, 13(20): 2924
7. Suaréz Concepción F., Piñero Aguilar R, Prieto Moreno A.S., Alfonso Cordoví A., Carbó Castro J.C., Llanes-Santiago O. Metodología para la automatización de procesos tecnológicos en la industria farmacéutica cubana. Ingeniería Industrial. 2022, 43(1): 1-14
8. Arden N. S.; Fisher A. C., Tyner K.; Yu L. X., Lee S. L., Kopcha M. Industry 4.0 for pharmaceutical manufacturing: Preparing for the smart factories of the future. International Journal of Pharmaceutics. 2021, 602: 1-8
9. Varghese B., Wang N., Barbhuiya S., Kilpatrick P., Nikolopoulos D. S. Challenges and Opportunities in Edge Computing. IEEE International Conference on Smart Cloud. 2016. 18: 20-26
10. Shi W., Cao J., Zhang Q., Li Y., Xu L. Edge Computing: Vision and Challenges. IEEE INTERNET OF THINGS JOURNAL. 2016. 3(5): 637-646
11. Pérez Colomé, A. L.; Anís Calderón, C.; Delgado Fernández, T. Procedimiento para la implementación de la computación en la niebla en ciudades inteligentes. Revista de Ingeniería Electrónica, Automática y Comunicaciones. 2021, 41(1): 45-57.
12. Shehab A. H.; Al-Janabi S. Edge Computing: Review and Future Directions (Computación de Borde: Revisión y Direcciones Futuras) Arquitectura/Urbanismo/Sustentabilidad. 2019; 26(2): 368-380.
13. Ali Y. Edge Computing Paradigms: Bridging the Gap between Software Engineering and IoT. Journal of Electrical Systems. 2024, 20(6s) 1381-1393.
14. Musarrat Z., Mysun M., Most Marufatul J.M., Md Motaharul I., Md Rafiul H. Mohammad M.H., Energy-Efficient Architecture for Optimized IoT Data Transmission from Edge to Cloud. ResearchGate [database on the Internet], 2024. Disponible en: doi: https://doi.org/10.21203/rs.3.rs-4127989/v1

15. Sittón-Candanedo I., Alonso R., Rodríguez S., Garcia Coria J., De La Prieta F. Edge Computing Architectures in Industry 4.0: A General Survey and Comparison. ResearchGate [database on the Internet], 2020. Disponible en: doi: 10.1007/978-3-030-20055-8_12
16. Sittón-Candanedo I, Alonso R. S, Múñoz L, Rodríguez-González S. Arquitecturas de Referencia Edge Computing para la Industria 4.0: una revisión. Congreso Internacional en Inteligencia Ambiental, Ingeniería de Software y Salud Electrónica y Móvil AmITIC 2019. Pereira, Colombia, 2019. p. 16-23 https://revistas.utp.ac.pa/index.php/memoutp/article/view/2284
17. Cao K., Liu Y., Meng G. Sun Q., An Overview on Edge Computing Research. IEEE Access, 2020, 8: 85714-85728, doi: 10.1109/ACCESS.2020.2991734.
18. Venanzi R., Solimando M., Patrali M., Foschini L., ChatzimisiosP. Siemens and EdgeX IIoT Platforms: A Functional and Performance Evaluation. ICC 2023 - IEEE International Conference on Communications, Rome, Italy; 2023. p. 834-839, doi: 10.1109/ICC45041.2023.10278750
19. Instrument Society of America, ANSI/ISA–S95.00.03–2013 Enterprise-Control System Integration Part 3: Activity Models of Manufacturing Operations Management. Research Triangle Park, North Carolina (USA); 2013
20. Instrument Society of America, ANSI/ISA–S95.00.01–2000 Enterprise-Control System Integration Part 1: Models and Terminology. Research Triangle Park, North Carolina (USA); 2000
21. Ruíz Zamora, J. F. Un análisis sobre mecanismos de comunicación entre componentes software. Máster en Tecnologías y Aplicaciones en Ingeniería Informática, Universidad de Almería; 2022. Disponible en: https://repositorio.ual.es/bitstream/handle/10835/16748/RUIZ%20ZAMORA,%20JOSE%20FRANCISCO.pdf?sequence=1
22. Khononov, V. Learning Domain-Driven Design. 1ra ed. O'Reilly Media, Inc; 2021.
23. Kamiński, L.; Kozłowski, M.; Sporysz, D.; Wolska, K.; Zaniewski, P.; Roszczyk, R. Comparative Review of Selected Internet Communication Protocols. Foundations of Computing and Decision Sciences. 2023, 48(1): 39-56

CONFLICTO DE INTERESES

Ninguno de los autores manifestó la existencia de posibles conflictos de intereses que debieran ser declarados en relación con este artículo.

CONTRIBUCIONES DE LOS AUTORES

- **Conceptualización de la arquitectura**: Zuliet Medina Rodríguez, Alberto Sergio Prieto Moreno y Hernán David Pérez Villanueva
- **Investigación:** Zuliet Medina Rodríguez, Alberto Sergio Prieto Moreno
- **Metodología para la realización de la arquitectura:** Zuliet Medina Rodríguez, Alberto Sergio Prieto Moreno y Hernán David Pérez Villanueva
- **Administración de proyecto:** Alberto Sergio Prieto Moreno
- **Búsqueda de recursos informáticos para la implementación:** Zuliet Medina Rodríguez, Alberto Sergio Prieto Moreno y Hernán David Pérez Villanueva
- **Programación, desarrollo de software; diseño de programas informáticos; implementación del código y prueba de componentes de código existentes:** Zuliet Medina Rodríguez
- **Supervisión:** Alberto Sergio Prieto Moreno y Hernán David Pérez Villanueva
- **Validación –Verificación de los componentes actualmente programados:** Zuliet Medina Rodríguez
- **Redacción- borrador original:** Zuliet Medina Rodríguez
- **Redacción – revisión y edición:** Zuliet Medina Rodríguez, Alberto Sergio Prieto Moreno

AUTORES

Ing. Zuliet Medina Rodríguez. Graduada con título de oro de Ingeniera en Automática en julio del año 2023 de la Universidad Tecnológica de La Habana "José A. Echeverría" (CUJAE). Se desempeña como docente en la facultad de Ingeniería Automática y Biomédica, además forma parte del equipo de trabajo de la empresa EMSI FARMA S.R.L. como

Especialista en Informatización Industrial en La Habana, Cuba. Correo: zulietmr@gmail.com, ORCID: https://orcid.org/0000-0002-1972-5322. Entre sus principales temas de interés se encuentran: desarrollo de software (Backend) con tecnología de .Net, soluciones automatizadas, protocolos de comunicación y servicios y microservicios web.

Dr. Alberto Sergio Prieto Moreno. Graduado con título de oro de Ingeniera en Automática en julio del año 2004 de la Universidad Tecnológica de La Habana "José A. Echeverría" (CUJAE). Doctor en Ciencias Técnicas, de la propia universidad en el año 2013. Se desempeña como profesor Titular a tiempo parcial y subdirector de la empresa EMSI FARMA S.R.L. en La Habana, Cuba. Correo: albprietom@gmail.com, ORCID: https://orcid.org/0000-0001-9907-885X. Entre sus principales temas de interés se encuentran: análisis de datos, diagnóstico de fallos, desarrollo de sistemas para la industria empleando el paradigma I4.0 con tecnología .NET y microservicios web.

Ing. Hernán David Pérez Villanueva. Graduado de Ingeniería Automática en enero del 2022 de la Universidad Tecnológica de la Habana "José A. Echeverría" (CUJAE). Se desempeña como profesor adiestrado en el Departamento de Automática de dicha universidad y como Especialista en Informática Industrial en la empresa EMSE FARMA S.R.L. en La Habana, Cuba. Correo: hernandpv41@gmail.com, ORCID: https://orcid.org/0009-0003-5066-6556. Entre sus principales temas de interés se encuentran: desarrollo de sistemas para la industria empleando el paradigma I4.0 con tecnología .NET, microservicios web y procesamiento de imágenes.