ISSN: 1692-7257 - Volumen 1 – Número 43 - 2024

RCTA
Revista Colombiana de Tecnologías de Avanzada
UNIPAMPLONA

Digital Object Identifier: 10.24054/rcta.v1i43.2912

Patrón para identificar en la web de las cosas los puntos de despliegue de algoritmos de inteligencia artificial

Pattern for identifying the deployment points of artificial intelligence algorithms on the internet of things (IoT)

MSc. Camilo Enrique Romero Parra [1], PhD. Carlos Alberto Cobos Lozada [1]
PhD. Miguel Ángel Niño Zambrano [1]

[1] ***Universidad del Cauca,*** *Facultad de Ingeniería Electrónica y Telecomunicaciones, Grupo de investigación y desarrollo en tecnologías de la información - GTI, Popayán, Cauca, Colombia.*

Correspondencia: *ceromero@unicauca.edu.co*

Recibido: *8 noviembre 2023.* ***Aceptado:*** *10 enero 2024.* ***Publicado:*** *7 mayo 2024.*

Cómo citar: *C. E. Romero Parra, C. A. Cobos Lozada, y M. A. Niño Zambrano, «Patrón para identificar en la web de las cosas los puntos de despliegue de algoritmos de inteligencia artificial», RCTA, vol. 1, n.º 43, pp. 144–154, may 2024. Recuperado de https://ojs.unipamplona.edu.co/index.php/rcta/article/view/2912*

Resumen: Al desarrollar soluciones IoT se presentan limitaciones en las capacidades de hardware y software. Además, el desarrollador deberá seleccionar la ubicación dentro del ecosistema que mejor se adecua a las necesidades del desarrollo, teniendo tres posibles ubicaciones para el procesamiento: el borde de la red, la niebla y la nube. El presente artículo propone un patrón arquitectónico que permite guiar la selección del punto de despliegue de aplicaciones basadas en algoritmos de inteligencia artificial con base en las necesidades de la solución tecnológica desarrollada. El patrón se obtuvo utilizando como metodología el Patrón de Investigación Iterativa de Pratt. Se usó un ejemplo del mundo real y se aplicó el paso a paso propuesto para demostrar la utilidad del patrón. Se concluyo que la selección de la ubicación del procesamiento debe tener en cuenta las necesidades del usuario final y las limitaciones que se puedan presentar.

Palabras clave: Internet de las Cosas, Patrón Arquitectónico, Web de las Cosas, Ingeniería de software, Inteligencia Artificial.

Abstract: When developing IoT solutions, limitations in hardware and software capabilities arise. Additionally, the developer must choose the location within the ecosystem that best suits the development needs, having three possible processing locations: the network edge, the fog, and the cloud. This article proposes an architectural pattern that guides the selection of deployment point for AI algorithm-based applications based on the needs of the developed technological solution. The pattern was obtained using the Pratt's Iterative Research Pattern as a methodology. A real-world example was used, and the proposed step-by-step approach was applied to demonstrate the pattern's usefulness. It was concluded that the selection of processing location should consider the end-user needs and any limitations that may arise.

Keywords: Internet of Things, Architectural Pattern, Web of Things, Software Engineering, Artificial Intelligence.

1. INTRODUCCIÓN

En años recientes se ha incrementado el interés por el desarrollo de soluciones en ecosistemas IoT, compuestos por objetos inteligentes que conectan entre sí para crear servicios interoperables. Uno de los principales obstáculos en la creación de estos sistemas es la alta heterogeneidad, es decir, la enorme cantidad de tecnologías y protocolos utilizados por los diferentes fabricantes [5].

Adicionalmente, el desarrollador de una solución en este entorno debe seleccionar una ubicación para realizar el procesamiento (y análisis) de la información suministrada por los objetos inteligentes. Dependiendo de la ubicación seleccionada la aplicación podrá contar con distintas ventajas y desventajas.

Existen tres puntos de despliegue para realizar el procesamiento, el borde (Edge), la niebla (Fog) y la nube (Cloud) [6][7]. El Edge puede procesar los datos adquiridos en el punto de origen siempre y cuando la aplicación IoT tenga las capacidades de procesamiento y comunicación en el propio dispositivo inteligente [8]. En el Fog hay un único dispositivo centralizado responsable de procesar (y analizar) los datos de diferentes puntos finales en la red, es decir, toma los datos de los dispositivos inteligentes ubicados en el Edge [7]. Y en la nube se cuenta con servidores con las mayores capacidades de procesamiento, almacenamiento, y autenticación de usuario a los cuales se deben reportar los datos desde los diferentes puntos de captura [6].

La principal ventaja que presenta la computación en el Edge frente a la computación en el Fog y en el Cloud está en los tiempos de respuesta, los cuales pueden reducirse gracias a que el procesamiento se hace de manera local tan pronto como se adquieren los datos por parte del dispositivo inteligente, mientras que su principal desventaja está en las capacidades limitadas del hardware en comparación con los servidores alojados en el Cloud. Por otro lado, la ventaja que presenta la computación en el Cloud frente a la computación en el Fog está en la capacidad de procesamiento y almacenamiento de datos, teniendo como desventaja una mayor dificultad para acceder a los datos suministrados por los objetos inteligentes [9]. Dado lo anterior, los diseñadores de aplicaciones IoT tienen inconvenientes al momento de decidir en qué sitio de la red desplegar los diferentes algoritmos de procesamiento de datos. Normalmente toman la decisión de manera empírica, lo cual en muchos casos genera retrasos en el desarrollo de las soluciones, o en el peor de los casos, aplicaciones que no cumplen del todo con los requisitos funcionales o de eficiencia [10] [11]; estas últimas de especial cuidado en las aplicaciones IoT que en la mayoría de los casos son soluciones que deben actuar en tiempo real, tener un bajo consumo de memoria, realizar la justa cantidad de procesamiento para ahorrar batería y usar poco ancho de banda.

Por esta razón, es importante encontrar una herramienta que apoye a los desarrolladores de aplicaciones IoT para que tomen la mejor decisión en la definición de la ubicación del despliegue de los algoritmos de procesamiento (incluido el análisis de datos) dentro de un Ecosistema de Objetos Inteligentes de la Web de las Cosas (Intelligent Objects Ecosystem of Web of Things - IOEoWoT), buscando aprovechar los beneficios propuestos por las tres ubicaciones previamente mencionadas.

En este contexto, el presente estudio propone un patrón arquitectónico que permite definir el punto de despliegue de un algoritmo de inteligencia computacional en una solución que se enmarca en un ecosistema de objetos inteligentes de la web de las cosas, con el fin de orientar al desarrollador en la solución que mejor se ajuste a los requerimientos de la aplicación IoT.

El artículo se encuentra organizado de la siguiente manera: La sección 2 describe la metodología que se usó, la cual se basó en Patrón de Investigación Iterativa propuesto por Pratt [12], la sección 3 introduce el concepto de patrón arquitectónico para la IoT y luego detallar el patrón arquitectónico, la sección 4 presenta un ejemplo de uso del patrón y la sección 5 la discusión, conclusiones y trabajos futuro que el grupo de investigación espera desarrollar en el corto plazo sobre el tema.

2. METODOLOGÍA

El patrón arquitectónico que se propone en este documento se obtuvo utilizando como metodología el Patrón de Investigación Iterativa (PII) de Pratt [12]. El PII se destaca por ser un proceso iterativo e incremental en el que se desarrollan cuatro etapas en cada iteración, la observación del problema, la identificación del problema, el desarrollo de una solución al problema y la evaluación de la solución desarrollada. Para la definición del patrón propuesto se requirieron tres iteraciones, en las que se fue

aprendiendo y cambiando la solución propuesta hasta obtener la versión que se presenta en la siguiente sección de este documento.

En la primera iteración se buscó comparar experimentalmente el rendimiento obtenido al ejecutar algoritmos de inteligencia artificial (algoritmos de lógica difusa, algoritmos genéticos, entre otras pruebas) en diferentes ubicaciones de ecosistemas inteligentes en la IoT, realizando pruebas entre dispositivos con distintas capacidades de hardware en cada ubicación (por ejemplo, en el Edge se usó una tarjeta ESP8266 como el dispositivo de menor capacidad de cómputo y la tarjeta Raspberry Pi 4 como el dispositivo con mayor capacidad de cómputo). Se observó que, para guiar la elección de la ubicación sería conveniente presentar al usuario una comparativa con el tiempo de respuesta obtenido al ejecutar un mismo algoritmo en diferentes ubicaciones, así como identificar cuales dispositivos no cuentan con las capacidades necesarias para llevar a cabo el procesamiento de algoritmos concretos. Se identificó como problema de investigación la gran cantidad y variedad de dispositivos disponibles para realizar los experimentos planteados, y la forma como se deben implementar las soluciones de inteligencia artificial dependiendo de las necesidades del usuario. Luego de realizar una comparativa de rendimiento entre dispositivos en distintas ubicaciones con distintos algoritmos de inteligencia artificial, se llegó a la conclusión de que era necesario un cambio en la forma como se estaba desarrollando el proyecto ya que el enfoque experimental no era viable debido a que esto requeriría implementar un número indeterminado y muy grande de pruebas que en el tiempo podría seguir creciendo en cantidad y variedad. Igualmente, se evidenció la poca literatura con la que se cuenta para implementar algoritmos de inteligencia computacional en las tarjetas basadas en Arduino.

En la segunda iteración se cambió el enfoque experimental y se buscó una solución general para guiar la ubicación del procesamiento dependiendo de las necesidades del usuario. Se observó que para seleccionar la ubicación se debían identificar los requisitos funcionales y no funcionales de la aplicación que se quisiera realizar (tiempo de respuesta, efectividad, disponibilidad de los datos adquiridos, capacidad de almacenamiento y procesamiento, seguridad, y escalabilidad). Se identificó como problema de investigación, la necesidad del usuario por delimitar las posibilidades de ubicación para las aplicaciones que deseara desarrollar y como seleccionar la mejor ubicación dentro de las posibles soluciones. En este sentido, se planteó describir los escenarios de uso del patrón y sus requisitos de uso. Tomando como referente las características de un patrón de diseño definidas por Gamma et. Al [13], junto con las características mencionadas por Bloom et. Al [14], se plantearon las características del patrón y se creó un borrador del patrón en el cual se estipulan las condiciones que debe cumplir el usuario para utilizar el patrón y se guía el desarrollo con base en la metodología CRISP-DM [15]. Al valorar el patrón que se desarrolló hasta ese momento con un experto, se observó qué era necesario mejorar su nivel de detalle, con el fin de proporcionar una guía que ofreciera mayor claridad sobre las ubicaciones que son adecuadas para desplegar el procesamiento y que permitiera elegir cual será la más adecuada para cada escenario de uso.

En la tercera iteración se creó la versión final del patrón arquitectónico de ubicación del procesamiento en ecosistemas IoT, se describieron sus componentes y se realizaron los ejemplos para comprobar su utilidad.

3. PATRÓN ARQUITECTÓNICO PROPUESTO

A continuación, se introduce el concepto de patrón arquitectónico para la IoT y luego se presentan los componentes del patrón arquitectónico de ubicación del procesamiento propuesto, componentes que se organizan en contexto, problema y solución.

3.1. Patrones arquitectónicos para la IoT

Para que una solución sea considerada un patrón debe capturar una práctica común (lo que implica que debe tener al menos tres usos conocidos) y, al mismo tiempo, la solución del patrón no debe ser obvia. La descripción de los patrones arquitectónicos normalmente se basa en la tripleta contexto-problema-solución, y estos funcionan de forma sinérgica en el contexto de un lenguaje de patrones con numerosas interdependencias con otros patrones [16].

Dentro del campo de la IoT existen diferentes categorizaciones para los patrones, como se puede observar en el artículo propuesto por Washizaki [17], dónde realizan una revisión sistemática de la literatura en la cual buscan describir de manera general el panorama actual de los patrones de diseño y de arquitectura de la IoT, para identificar

deficiencias y sugerir mejoras a la hora de crear nuevos patrones. Dentro de este mismo artículo, se realiza una clasificación de los patrones dependiendo del nivel de abstracción, especificidad del dominio, y requisito no funcional a abordar.

Siguiendo la categorización propuesta por Washizaki, se determinó que: 1) el nivel de abstracción del patrón propuesto es medio, buscando garantizar la interoperabilidad entre dispositivos heterogéneos, conociendo el contexto de las necesidades del desarrollo, problemas recurrentes, y su correspondiente solución; 2) la especificidad de dominio del patrón propuesto corresponde a general, ya que es aplicable a cualquier sistema IoT; y 3) Los requisitos no funcionales que tiene en cuenta el patrón propuesto fueron tiempo de respuesta, despliegue y actualización del modelo o del algoritmo, adquisición de los datos, procesamiento, seguridad, movilidad, consumo energético y costo del despliegue.

Por otro lado, el patrón propuesto incluye todos los enfoques de la inteligencia artificial, a saber: sistemas difusos, redes neuronales, metaheurísticas, aprendizaje de máquina y sistemas expertos [18].

3.2. Contexto

El patrón está dirigido a desarrolladores que implementan una aplicación IoT y busca orientar la elección de la ubicación del procesamiento (incluido el análisis de datos) utilizando algoritmos de inteligencia artificial en la nube (Cloud), en la niebla (Fog) o en el borde (Edge). Es preciso comentar que se requiere un nivel básico de experiencia del usuario del patrón para que responda correctamente (de acuerdo con las necesidades de la aplicación que este desarrollando) las preguntas que se deben resolver y que soportan la decisión de la ubicación de la aplicación/solución IoT que se está desarrollando.

Dentro de un ecosistema de objetos inteligentes de la web de las cosas, existen distintos requisitos no funcionales (seguridad, tiempo de respuesta, conectividad, escalabilidad, entre otros) a los cuales se puede enfocar el diseño de la solución que se quiera implementar. Dependiendo de este enfoque, se puede tomar la decisión de realizar el procesamiento en una ubicación dentro del ecosistema [19].

Por otra parte, dependiendo de la técnica de inteligencia artificial (machine learning, neural networks, expert systems, fuzzy logic y metaheuristics) que se necesite implementar para dar solución a un determinado problema, en una aplicación específica, es necesario el uso de componentes hardware con capacidades de procesamiento y almacenamiento mínimas para poder desempeñar la tarea requerida.

Dada la complejidad de lograr un equilibrio adecuado entre los requisitos no funcionales y los objetivos de procesamiento de la aplicación particular, este patrón apoya la toma de la decisión de la ubicación más apropiada (Edge, Fog o Cloud), teniendo en cuenta las necesidades específicas de la aplicación IoT que se quiere implementar.

Dependiendo de la ubicación del procesamiento dentro de un ecosistema de objetos inteligentes, se obtendrán ventajas y desventajas desde el punto de vista del usuario final. En el Edge la velocidad de respuesta es mayor, se tienen mayores capacidades de movilidad, autonomía energética, entre otros. En el Fog se tienen capacidades superiores de autenticación de usuario, lo cual influye en la seguridad de la aplicación, capacidades de almacenamiento superiores, capacidad para concentrar datos de dispositivos inteligentes que están en el Edge, entre otras. Finalmente, en el Cloud existe una mayor capacidad de procesamiento, así como de optimización y simulación [19]. En la Tabla 1 se muestra a manera de ejemplo una comparativa entre las tres posibles ubicaciones con algunos requisitos no funcionales.

Un ejemplo de un escenario de motivación para hacer uso del presente patrón, podría ser una aplicación IoT enfocada en la traducción del lenguaje de señas a texto mediante visión de máquina y algoritmos de inteligencia artificial, más específicamente mediante una red neuronal. En este ejemplo, se presenta una dificultad para el diseñador de la aplicación a la hora de seleccionar la mejor ubicación para el procesamiento de los datos adquiridos por el sensor (en este caso las posiciones de las manos capturadas con la cámara), dado que la captura de datos y el entrenamiento de la red neuronal pueden tomar mucho más tiempo (en caso de que sea del todo posible) en un dispositivo ubicado en el Edge (cómo puede ser una tarjeta Arduino) en comparación con un dispositivo en otra ubicación. Adicionalmente, en el Cloud se necesitará de una conexión a internet para consumir los servicios de procesamiento necesarios con la ventaja de tener una mayor velocidad en el procesamiento y una posible desventaja en la latencia de la conexión.aterial gráfico debe llamarse

directamente en el texto o entre paréntesis, como por ejemplo "(ver Fig. 1)". Este material debe presentarse de manera independiente, numerándolo consecutivamente (Fig. 1, Fig. 2, etc.) y proporcionando el título y la fuente correspondiente.

Tabla 1: *Ventajas y desventajas de las ubicaciones*

Requisito	Cloud	Fog	Edge
Latencia	Desventaja	Neutral	Ventaja
Distancia entre la ubicación y los dispositivos	Desventaja (Lejos del borde)	Neutral (Cerca al borde)	Ventaja
Almacenamiento	Ventaja	Neutral	Desventaja
Potencia de computo	Ventaja	Neutral (depende del dispositivo)	Desventaja
Capacidad de movilidad	Desventaja	Neutral	Ventaja
Autenticación de usuario	Ventaja	Neutral	Desventaja

Fuente: *elaboración propia*

3.3. Problema

¿Dónde ubicar una aplicación software basada en inteligencia artificial en una arquitectura que contempla el Edge, Fog y Cloud en un ecosistema de objetos inteligentes de la WoT?

3.4. Solución

Antes de definir la ubicación del algoritmo para su uso (Edge, Fog o Cloud), el desarrollador deberá realizar el entrenamiento del algoritmo. En el caso de las soluciones basadas en sistemas expertos, lógica difusa y metaheurísticas, a pesar de que no se tiene una fase con el nombre de entrenamiento, el desarrollador tiene que construir las reglas o algoritmos con base en la experiencia de los expertos del dominio o el problema concreto que se está resolviendo, que es similar a un entrenamiento de un modelo pero que no se hace de forma automática desde datos. El entrenamiento normalmente se realiza en la ubicación con las mayores capacidades de procesamiento disponibles, donde se cuenta además con las herramientas necesarias para su desarrollo. El desarrollador deberá tener muy en cuenta que el modelo obtenido del entrenamiento, en el caso de este ejemplo una red neuronal, pueda ubicarse en el lugar de despliegue seleccionado.

Una vez se haya realizado el entrenamiento previo del algoritmo, el usuario (desarrollador o arquitecto de la solución) debe evaluar la viabilidad (factibilidad) de cada ubicación (Edge, Fog o Cloud) para la implementación de la solución. Luego, si dos o más ubicaciones se consideran viables se debe evaluar de acuerdo con el contexto de la solución cuál de las ubicaciones es la mejor alternativa.

Dado lo anterior, se propuso un conjunto de preguntas que definen la viabilidad (factibilidad) de ubicar el procesamiento en una determinada ubicación, como se puede observar en la Tabla 2 (ver Anexos). Las preguntas deberán ser respondidas con SI o NO. Se inicia evaluando la viabilidad de ubicar la solución en el Edge, en caso de que una de las respuestas sea negativa, esta ubicación será definida como inviable para realizar el procesamiento. Este proceso se repite luego para la ubicación en el Fog y después para el Cloud.

Una vez definidas las ubicaciones viables, se procede a determina cuál de ellas es la mejor alternativa. Para lograr esto se deben primero ponderar los requisitos no funcionales de la aplicación (tiempo de respuesta, latencia, escalabilidad, facilidad de entrenamiento del modelo de inteligencia artificial y su actualización, facilidad de adquisición de datos, capacidad de procesamiento, interoperabilidad, autonomía y seguridad) de la solución que se va a implementar. La Tabla 3 (ver Anexos) muestra en su primera columna los requisitos no funcionales para tener en cuenta y en la segunda su importancia correspondiente, la cual deberá puntuarse sin tener en cuenta la ubicación en la que será desplegada la aplicación. La importancia es una evaluación subjetiva, que depende de los requisitos identificados de la solución IoT que se va a desarrollar. Depende de la experiencia del desarrollador y del ingeniero de requisitos valorar correctamente esta importancia.

Posteriormente, en la Tabla 4 (ver Anexos) se presenta la evaluación de los requisitos no funcionales. En esta tabla se deberá copiar la misma puntuación de la importancia de los requisitos que se asignó previamente en la Tabla 3 y se dará una calificación a cada una de las preguntas completas que aparecen en la primera columna (según la ubicación que se desee evaluar) con un valor en una escala de 0 a 5 (donde 0 se refiere al peor desempeño y 5 al mejor desempeño) con base en los resultados que se espera provea el algoritmo al ser ejecutado en una determinada ubicación, esta calificación se asigna con base en la experiencia previa del desarrollador de la aplicación (que se puede adquirir haciendo experimentos sobre los dispositivos en las ubicaciones o consultando con expertos) o con base en reportes encontrados en la literatura, lo cual en los dos casos establece una calificación estimada.

Una vez se tiene la puntuación de cada ubicación viable, se multiplica la ponderación (importancia) de cada requisito no funcional por su calificación, se suman dichos valores y al final se divide por la suma de las ponderaciones para obtener la calificación final de cada ubicación. La ubicación que obtenga la puntuación más alta se selecciona como la ubicación más apropiada (1).

$$Decisión = ArgMax_{u_J \in U}\left(\sum_{r=1}^{8}(I_r \times Cal_{r,u_J})\Big/\sum_{r=1}^{8} I_r\right) \quad (1)$$

Donde, U corresponde a las ubicaciones (Edge, Fog o Cloud), u_J corresponde a cada una de las ubicaciones, r Índice de cada uno de los 8 requisitos no funcionales, I_r Importancia dada a un requisito no funcional según la Tabla 3, Cal_{r,u_J} Calificación del requisito no funcional r en la ubicación u_J según la Tabla 4 (ver Anexos).

A continuación, se explican en detalle los requisitos no funcionales:

Tiempo de respuesta: Es el tiempo resultante de la suma entre el tiempo de adquisición de los datos, procesamiento, entrega de respuesta y latencia. Para valorar este requisito en una determinada ubicación el desarrollador deberá preguntarse si con los dispositivos que tiene disponibles, con el alcance que tienen estos dispositivos, puede cumplirse con un tiempo de respuesta apropiado para el usuario.

Despliegue y actualización del modelo o del algoritmo: Si la aplicación que se va a ubicar se basa en un modelo que además de desplegarse se debe actualizar con una cierta periodicidad, se debe tener claro que esta tarea se puede realizar en la ubicación seleccionada (eficacia), pero, además, que el desarrollo de estos dos procesos se pueda realizar con recursos software, hardware, tiempo y costo, que está en concordancia con el presupuesto del proyecto o de la empresa (eficiencia).

Adquisición de los datos: Es una calificación estimada de que tan eficiente es la recolección (adquirir) de los datos requeridos para la ejecución de la aplicación basada en IA. Evaluar esta eficiencia puede incluir el tiempo de recolección desde las diferentes fuentes y el consumo energético con los diferentes recursos disponibles en la ubicación.

Procesamiento: La capacidad de procesamiento disponible (procesador, RAM y almacenamiento externo) en la ubicación permitirá obtener el resultado de la ejecución del modelo o el algoritmo en un tiempo que está dentro del rango de lo esperado por parte del usuario.

Seguridad: La ubicación donde se va a alojar la aplicación cuenta con dispositivos que vienen con herramientas de seguridad integradas, estas están apropiadamente configuradas, las contraseñas para el acceso a los dispositivos están inicializadas y son robustas, la definición de la red protege del acceso no autorizado a los dispositivos, los dispositivos tienen su firmware, sistema operativo y otras aplicaciones actualizadas, protegidas y se actualizan constantemente, las funcionalidades o características que no se usan en los dispositivos están deshabilitadas; con lo que se logra que la aplicación este instalada en un entorno donde la seguridad tiene el nivel requerido por la organización que la va a usar.

Movilidad: El usuario cuenta con la suficiente capacidad de desplazamiento en la ubicación seleccionada para el uso de la aplicación y el desarrollo de la tarea para la cual esta fue diseñada.
Consumo energético: La aplicación al ser instalada en la ubicación seleccionada, contará con la capacidad energética suficiente para ser usada todo el tiempo que sea necesario, o por el contrario necesita baterías u otros mecanismos de consumo energético.

Costo: La compra, reposición o alquiler de equipos y software actualizados en la ubicación seleccionada corresponden con el presupuesto establecido para el despliegue de la aplicación.

4. EJEMPLO DE USO DEL PATRÓN PROPUESTO

A continuación, se presenta a manera de ejemplo, la decisión de ubicar una aplicación que realiza el procesamiento de imágenes para la selección automática de piezas defectuosas en una línea de producción usando el patrón propuesto. En ese ejemplo se cuenta con una cámara (dispositivo inteligente) que transmite las imágenes capturadas a un computador (dispositivo concentrador) con un sistema operativo actualizado con parches de seguridad, con procesador Intel Core i5 de 8va generación, sin tarjeta gráfica, con 4GB de RAM, en una fábrica, la cual cuenta con servicios de procesamiento en el Cloud para su intranet y extranet. Los desarrolladores consideran que la aplicación se debe desarrollar con una red neuronal profunda.

Siguiendo las indicaciones de la solución, el desarrollador deberá realizar un entrenamiento con el fin de obtener el modelo de la red neuronal a implementar. Una vez se cuenta con un modelo, que en este caso será compatible con las tres posibles ubicaciones, se pasa a evaluar la viabilidad de ubicar el procesamiento en esas tres ubicaciones. A continuación, la Tabla 5 lista los datos requeridos para la aplicación, así como el lugar donde se alojan estos datos. Esta información se mantiene para las tres posibles ubicaciones. Las preguntas orientadoras y las acciones para contestar la pregunta se muestran en la Tabla 2.

A continuación, en la **¡Error! No se encuentra el origen de la referencia.** (ver Anexos), se realiza la evaluación de la viabilidad del despliegue de la aplicación en el Edge. Como se puede apreciar, esta ubicación no es viable y por lo tanto se descarta.

A continuación, en la Tabla 7 (ver Anexos), se realiza la evaluación de la viabilidad del despliegue de la aplicación en el Fog. Esta ubicación es viable, y por lo tanto pasa a la fase de calificación de requisitos no funcionales.

A continuación, en la Tabla 8 (ver Anexos), se realiza la evaluación de la viabilidad del despliegue de la aplicación en el Cloud. Esta ubicación es viable, y por lo tanto pasa a la fase de calificación de requisitos no funcionales.

De esta manera se sabe que las ubicaciones viables son el Fog y el Cloud. Para lograr determinar cuál de estas ubicaciones será la mejor alternativa para ubicar el procesamiento de la aplicación IoT se utilizan las Tabla 3 y Tabla 4. La evaluación de la importancia se muestra a continuación en la Tabla 9 (ver Anexos).

El resultado de la evaluación en el Fog del ejemplo se muestra a continuación en la Tabla 10 (ver Anexos). Como se puede apreciar, la calificación no se ajusta con la importancia del procesamiento, sin embargo, los demás requisitos no funcionales se ajustan adecuadamente.

El resultado de la evaluación en el Cloud del ejemplo se muestra a continuación en la Tabla 11. Como se puede apreciar, el despliegue y actualización del modelo o del algoritmo y la adquisición de los datos no se ajustan con la valoración de la importancia que se requiere para estos requisitos no funcionales.

Finalmente, Se obtiene una calificación de 4,38 puntos para el Fog, superando la calificación de 3,77 puntos del Cloud, lo que indica que para la aplicación que se desea implementar el Fog es la mejor alternativa.

En la evaluación del patrón se desarrollaron dos pruebas de concepto las cuales fueron presentados a un grupo focal compuesto por expertos en metodologías agiles y expertos en IoT, metodologías agiles de desarrollo, y programación de entornos en IoT; sin embargo, estas no se incluyen en el presente artículo para no sobrepasar la longitud límite definida por la revista. En caso de ser requeridas pueden ser solicitadas vía correo electrónico con el autor de correspondencia.

5. DISCUSIÓN, CONCLUSIONES Y TRABAJO FUTURO

Durante el desarrollo del patrón fue posible entender que la selección de la ubicación del procesamiento (incluido el análisis de datos) no se limita a las capacidades hardware con las que cuenta una ubicación, sino que se hace necesario analizar las necesidades del usuario final y las limitaciones que se puedan presentar respecto al tiempo de respuesta, despliegue y actualización del modelo o del algoritmo, adquisición de los datos, procesamiento, seguridad, movilidad, consumo energético y costo.

El patrón propuesto en este estudio ha mostrado ser útil para guiar la elección del procesamiento en ecosistemas inteligentes cuando se hace uso de algoritmos de inteligencia artificial, como se pudo evidenciar en el ejemplo realizado.

Basado en las limitaciones que se identificaron del patrón propuesto, a continuación, se presentan los trabajos futuros a desarrollar:

- Comparar si la ubicación seleccionada utilizando el patrón propuesto corresponde con la ubicación seleccionada de manera experimental.
- Ampliar el alcance del patrón para guiar la ubicación del procesamiento de múltiples algoritmos o múltiples funcionalidades dentro de una misma solución.
- Agregar otros requisitos no funcionales dentro de una nueva versión del patrón tales como: escalabilidad, interoperabilidad, portabilidad, otros aspectos de mantenibilidad, actuación (el sistema debe manejar el numero requerido de usuarios sin que se degrade el rendimiento),

disponibilidad y compatibilidad. También incluir el análisis del contexto semántico en la IoT.

5. AGRADECIMIENTOS

A la Universidad del Cauca por financiar parcialmente la investigación.

REFERENCIAS

[1] P. Sethi and S. R. Sarangi, "Internet of Things: Architectures, Protocols, and Applications," *J. Electr. Comput. Eng.*, vol. 2017, p. 9324035, 2017, doi: 10.1155/2017/9324035.

[2] G. Bai, L. Yan, L. Gu, Y. Guo, and X. Chen, "Context-aware usage control for web of things," *Secur. Commun. Networks*, vol. 7, no. 12, pp. 2696–2712, 2014, doi: 10.1002/sec.424.

[3] A. Wagner, J. L. V. Barbosa, and D. N. F. Barbosa, "A model for profile management applied to ubiquitous learning environments," *Expert Syst. Appl.*, vol. 41, no. 4 PART 2, pp. 2023–2034, Mar. 2014, doi: 10.1016/j.eswa.2013.08.098.

[4] L. Yao, "A Propagation Model for Integrating Web of Things and Social Networks," in *Service-Oriented Computing - ICSOC 2011 Workshops*, G. Pallis, M. Jmaiel, A. Charfi, S. Graupner, Y. Karabulut, S. Guinea, F. Rosenberg, Q. Z. Sheng, C. Pautasso, and S. Ben Mokhtar, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 233–238.

[5] M. Niño Zambrano, "Interacción Semántica de Objetos en la Web de las Cosas," 2013.

[6] N. B. Ruparelia, *Cloud computing*. Mit Press, 2016. [Online]. Available: https://books.google.com.co/books?hl=es&lr=&id=umIsDAAAQBAJ&oi=fnd&pg=PR5&dq=Definition+of+Cloud+Computing&ots=k8BtuSpQ29&sig=FyIvnPN0V2lO12qQlPRrSekcaMk&redir_esc=y#v=onepage&q=Definition of Cloud Computing&f=false

[7] Z. Mahmood and M. Ramachandran, "Fog Computing: Concepts, Principles and Related Paradigms," in *Fog Computing: Concepts, Frameworks and Technologies*, Z. Mahmood, Ed., Cham: Springer International Publishing, 2018, pp. 3–21. doi: 10.1007/978-3-319-94890-4_1.

[8] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile Edge Computing: A Survey," *IEEE Internet of Things Journal*, vol. 5, no. 1. Institute of Electrical and Electronics Engineers Inc., pp. 450–465, Feb. 2018. doi: 10.1109/JIOT.2017.2750180.

[9] C. Mechalikh, H. Taktak, and F. Moussa, "A Scalable and Adaptive Tasks Orchestration Platform for IoT," in *2019 15th International Wireless Communications Mobile Computing Conference (IWCMC)*, 2019, pp. 1557–1563. doi: 10.1109/IWCMC.2019.8766744.

[10] B. Rababah, T. Alam, and R. Eskicioglu, "The Next Generation Internet of Things Architecture Towards Distributed Intelligence: Reviews, Applications, and Research Challenges," *SSRN Electron. J.*, 2020, doi: 10.2139/ssrn.3640136.

[11] M. Aazam, S. Zeadally, and K. A. Harras, "Offloading in fog computing for IoT: Review, enabling technologies, and research opportunities," *Futur. Gener. Comput. Syst.*, vol. 87, pp. 278–289, Oct. 2018, doi: 10.1016/J.FUTURE.2018.04.057.

[12] K. S. Pratt, "Design Patterns for Research Methods: Iterative Field Research," in *AAAI Spring Symp. Exp. Des. Real*, 2009, pp. 1–7.

[13] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, "Design Patterns: Abstraction and Reuse of Object-Oriented Design," 1993, pp. 406–431. doi: 10.1007/978-3-642-48354-7_15.

[14] G. Bloom, B. Alsulami, E. Nwafor, and I. C. Bertolotti, "Design patterns for the industrial Internet of Things," *IEEE Int. Work. Fact. Commun. Syst. - Proceedings, WFCS*, vol. 2018-June, pp. 1–10, Jul. 2018, doi: 10.1109/WFCS.2018.8402353.

[15] P. Chapman *et al.*, "CRISP-DM 1.0: Step-by-step data mining guide," 2000.

[16] P. Avgeriou and U. Zdun, "Architectural Patterns Revisited - A Pattern Language.," 2005, pp. 431–470.

[17] H. Washizaki, S. Ogata, A. Hazeyama, T. Okubo, E. B. Fernandez, and N. Yoshioka, "Landscape of Architecture and Design Patterns for IoT Systems," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 10091–10101, Oct. 2020, doi: 10.1109/JIOT.2020.3003528.

[18] A. Dounis, "Artificial intelligence for energy conservation in buildings," *Adv.*

Build. Energy Res., vol. 4, pp. 267–299, 2010, doi: 10.3763/aber.2009.0408.

[19] J. Queiroz, P. Leitão, J. Barbosa, and E. Oliveira, "Distributing Intelligence among Cloud, Fog and Edge in Industrial Cyber-physical Systems," in *ICINCO*, 2019.

[20] "(PDF) Internet of Things (IoT): Definitions, Challenges, and Recent Research Directions." https://www.researchgate.net/publication/320532203_Internet_of_Things_IoT_Definitions_Challenges_and_Recent_Research_Directions (accessed Nov. 27, 2022).

ANEXOS

Tabla 2: *Base de reglas*

Pregunta Orientadora	Acciones para contestar la pregunta
PO1: ¿Los datos requeridos para proveer el servicio se encuentran disponibles en la ubicación o pueden ser adquiridos desde está?	A1: Haga un listado de los datos y los atributos de estos que requieren para entrenamiento y uso de la aplicación (Este listado es igual para todas las ubicaciones, no lo repita).
	A2: Defina donde están cada uno de esos datos (Este listado es igual para todas las ubicaciones, no lo repita).
	A3: Asegúrese de que cada uno de los datos están disponibles desde la ubicación que se está evaluando como opción de despliegue.
PO2: ¿Existen herramientas para el desarrollo del procesamiento y análisis con inteligencia artificial, como frameworks o librerías con las implementaciones requeridas según el enfoque (machine learning, neural networks, expert systems, fuzzy logic o metaheuristics) de IA seleccionado?	A4: Basado en el tipo de implementación de IA, liste los frameworks o librerías que soportaran el despliegue de la solución, y que están disponibles en esta ubicación, que además cuenten con sus similares en el entorno de entrenamiento, desarrollo o modelado que se utilizara.
PO3: ¿El dispositivo cuenta con las capacidades hardware para realizar el procesamiento (procesador y RAM), almacenar los datos y conectarse con otras fuentes de datos si esto se requiere (red)?	A5: Haga un inventario de los dispositivos disponibles en esta ubicación que serán utilizados como puntos de despliegue de la implementación basada en IA. Si la empresa no cuenta con los dispositivos y los va a adquirir tendrá que hacer una revisión de cuáles son los que le brindan las mejores capacidades (procesamiento, memoria, almacenamiento, consumo, etc.) con las restricciones presupuestales de la empresa o proyecto.
	A6: Asegúrese de que cada uno de estos dispositivos tiene las capacidades (procesamiento, memoria, almacenamiento, red) que le permiten usar los frameworks y librerías disponibles para el despliegue de la aplicación.

Fuente: *elaboración propia*

Tabla 3: *Evaluación de la importancia por requisito*

Requisito no funcional	Importancia (0 menos importante y 5 más importante)
Tiempo de respuesta	
Despliegue y actualización del modelo o del algoritmo	
Adquisición de los datos	
Procesamiento	
Seguridad	
Movilidad	
Consumo energético	
Costo (se puntúa con base en los recursos disponibles)	

Fuente: *elaboración propia*

Tabla 4: *Evaluación de requisitos no funcionales*

Requisito no funcional con su pregunta asociada	Importancia	Calificación de la ubicación (Edge, Fog o Cloud) de 0 a 5	Calificación Ponderada
¿El tiempo de respuesta de la solución en esta ubicación es el apropiado para el desarrollador?			
¿El proceso de despliegue y actualización del modelo o del algoritmo en esta ubicación es eficiente y eficaz para el desarrollador?			
¿Se puede hacer la adquisición de los datos desde esta ubicación y aquí se puede desarrollar eficientemente?			
¿Con la capacidad de procesamiento disponible, el tiempo de respuesta es apropiado para la llevar a cabo la tarea?			
¿La seguridad que ofrece la ubicación es suficiente para cumplir con los requisitos de la aplicación?			
¿La capacidad de movilidad de los dispositivos en la ubicación satisfacen las			

necesidades del usuario? [20]			
¿El consumo energético de los dispositivos en la ubicación satisfacen las necesidades de la aplicación? [20]			
¿Qué tan viable es utilizar la ubicación según su costo de despliegue?			
TOTAL ACUMULADO	Suma de la calificación de importancias		Suma calificación ponderada
CALIFICACIÓN			Suma calificación ponderada / Suma de la calificación de importancias

Fuente: *elaboración propia*

Tabla 5: *Descripción de los datos*

PO1	
A1	Datos requeridos: Imágenes de los productos de la línea de producción. Datos requeridos para entrenamiento: Imágenes e identificadores de los productos con y sin defectos con las que se va a entrenar el modelo. Atributos requeridos: Nombre de la imagen con su estado (defectuoso o correcto) y dimensiones de la imagen.
A2	Los datos serán adquiridos mediante una fotografía, a la cual se le deberá dar el formato adecuado. Los datos de entrenamiento se obtienen de una base de datos de la empresa.

Fuente: *elaboración propia*

Tabla 6: *Evaluación de la viabilidad del despliegue en el Edge*

PO2	
A3	Si, los datos están disponibles en la ubicación.
PO3	
A4	Entorno de desarrollo integrado (IDE) para computador como VSCode o Pycharm (Utilizado para el entrenamiento del modelo). Entorno de desarrollo integrado para Raspberry Geany (Utilizado para el entrenamiento del modelo). Normalmente esta opción no se recomienda. Librerías: OpenCV, Scikit-image, PIL/pillow, NumPy, Mahotas
PO4	
A5	Dispositivo inteligente - cámara
A6	El dispositivo no cuenta con las capacidades hardware para realizar el procesamiento ni el almacenamiento de datos. No existen frameworks o librerías para el desarrollo en el sistema operativo propietario del fabricante del dispositivo.

Fuente: *elaboración propia*

Tabla 7: *Evaluación de la viabilidad del despliegue en el Fog*

PO2	
A3	Los datos no se encuentran en la ubicación, pero son fácilmente adquiribles al conectar la cámara al computador.
PO3	
A4	Entorno de desarrollo integrado (IDE) para computador como VSCode o Pycharm (Para entrenamiento y ejecución del modelo). Librerías: OpenCV, Scikit-image, PIL/pillow, NumPy, Mahotas
PO4	
A5	Computador de escritorio, con un sistema operativo actualizado con parches de seguridad, con procesador Intel Core i5 de 8va generación, sin tarjeta gráfica, con 4GB de RAM.
A6	El dispositivo cuenta con la capacidad hardware necesarias para llevar a cabo el procesamiento y almacenamiento de los datos, así como la capacidad para conectarse a distintos dispositivos de la red. Existen frameworks disponibles para desarrollar el algoritmo de procesamiento requerido.

Fuente: *elaboración propia*

Tabla 8: *Evaluación de la viabilidad del despliegue en el Cloud*

PO2	
A3	Si, los datos pueden ser enviados al Cloud pues se cuenta con conectividad a internet, por lo que se puede transferir la información desde el computador en el Fog.
PO3	
A4	Entorno de programación Google Colab, Pycharm, VSCode, etc. (Para entrenamiento y ejecución del modelo). Librerías: OpenCV, Scikit-image, PIL/pillow, NumPy, Mahotas.
PO4	
A5	Depende de los servidores contratados en la nube especifica (AWS, GCP, Azure, etc.), o hosting.
A6	La ubicación cuenta con las capacidades necesarias para realizar el procesamiento. Existen herramientas para llevar a cabo el procesamiento.

Fuente: *elaboración propia*

Tabla 9: *Evaluación de la importancia por requisito*

Requisito no funcional	Importancia
Tiempo de respuesta	4
Despliegue y actualización del modelo o del algoritmo	5
Adquisición de los datos	5
Procesamiento	4
Seguridad	1
Movilidad	1
Consumo energético	1
Costo	5

Fuente: *elaboración propia*

Tabla 10: *Ejemplo de evaluación de los requisitos no funcionales en el Fog*

Requisito no funcional con su pregunta asociada	**Importancia**	**Calificación de la ubicación (Edge, Fog o Cloud) de 0 a 5**	**Calificación Ponderada**

Tiempo de respuesta	4	5	20
Despliegue y actualización del modelo o del algoritmo	5	5	25
Adquisición de los datos	5	5	25
Procesamiento	4	3	12
Seguridad	1	5	5
Movilidad	1	1	1
Consumo energético	1	1	1
Costo	5	5	25
TOTAL ACUMULADO	26		114
CALIFICACIÓN			4,38

Fuente: *elaboración propia*

Tabla 11: *Ejemplo de evaluación de los requisitos no funcionales en el Cloud*

Requisito no funcional con su pregunta asociada	**Importancia**	**Calificación de la ubicación (Edge, Fog o Cloud) de 0 a 5**	**Calificación Ponderada**
Tiempo de respuesta	4	4	16
Despliegue y actualización del modelo o del algoritmo	5	3	15
Adquisición de los datos	5	3	15
Procesamiento	4	5	20
Seguridad	1	5	5
Movilidad	1	1	1
Consumo energético	1	1	1
Costo	5	5	25
TOTAL ACUMULADO	26		98
CALIFICACIÓN			3,77

Fuente: *elaboración propia*