Patrón para Identificar en la Web de las Cosas los Puntos de Despliegue de Algoritmos de Inteligencia Computacional

Trabajo de grado

Ing. Camilo Enrique Romero Parra

Anexo 4

Iteraciones uno y dos del patrón detalladas

Universidad del Cauca
Facultad de Ingeniería Electrónica y Telecomunicaciones
Programa de Maestría en Computación
Grupo de I+D en Tecnologías de la Información (GTI)

Popayán, febrero 2023

Primera iteración del patrón

A continuación, se presenta la primera iteración del patrón de investigación iterativa que se lleva a cabo para realizar la propuesta del patrón arquitectónico de ubicación del procesamiento.

Observar: Al observar la aplicación que se desea dar al patrón arquitectónico, se identifican necesidades específicas relacionadas con la efectividad de la posible solución para la aplicación IoT. En esta primera iteración se plantea la posibilidad de realizar una selección de la ubicación dependiendo de los tiempos de respuesta y el logro de los objetivos del usuario, y de manera general la posibilidad de ejecutar algoritmos de inteligencia computacional en distintos dispositivos ubicados en el borde de la red. Para ello, se realizarán pruebas de desempeño en distintas ubicaciones con distintos dispositivos, delimitando el escenario de uso del patrón.

Identificar los problemas de investigación: Dependiendo de los componentes hardware con los que se cuente dentro de la arquitectura, las posibilidades de realizar procesamiento en una determinada ubicación podrán variar, siendo de vital importancia realizar una categorización adecuada de los posibles componentes que puedan utilizarse dentro de la misma. Dependiendo del tipo de servicio que se quiera consumir dentro de la arquitectura, desde el punto de vista del usuario, se deberá aplicar un procesamiento específico que implemente un algoritmo de inteligencia computacional acorde con las necesidades de información. La decisión que se tome para ubicar el procesamiento dentro de la arquitectura dependerá tanto de las capacidades computacionales con las que cuentan las distintas ubicaciones, como con la complejidad del algoritmo aplicado y las características del problema que se desea resolver.

Desarrollar una solución al problema seleccionado: Se tomó la decisión de ejecutar pruebas de procesamiento de algoritmos, utilizando dos plataformas, las más representativas y utilizadas en los ecosistemas IoT, las cuales son, la tarjeta ESP8266 versión 0.1, con el fin de utilizar un hardware con características básicas y el un costo bajo en el mercado, junto con la tarjeta Raspberry Pi 4 modelo B del 2018, la cual fue escogida por ser un dispositivo de alta capacidad que sigue pudiendo ubicarse en el borde.

La tarjeta ESP8266 es un dispositivo con capacidad de conectividad wifi, cuenta con un microprocesador Tensilica L106 de 32 bits, voltaje de operación entre 2.5V y 3.6V, corriente de operación de 80 mA, rango de temperatura de funcionamiento de -40° C a 125° C y un tamaño de 5mm por 5mm.

La Raspberry Pi 4 modelo B cuenta con un procesador Quad core 64-bit ARM-Cortex A72 corriendo a 1.5GHz, 2 Gb de RAM LPDDR4, capacidad de conectividad wifi, Ethernet, USB y HDMI, con un almacenamiento de 16 Gb de memoria microSD.

La primera prueba consistió en realizar el montaje del framework Lazy Predict en la tarjeta Raspberry, el cual permite construir varios modelos básicos utilizando pocas

líneas de código, y ayuda a comprender qué modelos funcionan mejor sin ajustes de parámetros previos. La idea con esta prueba era realizar un benchmark (prueba de rendimiento) que se pudiera llevar a cabo en el borde, en la niebla, y en la nube. Como conclusión de esta primera prueba se determinó que la tarjeta Raspberry Pi 4 es capaz de ejecutar el algoritmo mencionado siempre y cuando estén disponibles las dependencias de paquetes para el framework así como para la arquitectura, ya que el programa no puede ser instalado de manera limpia como si se podría en un computador de escritorio o en otra plataforma que utilice el lenguaje Python. Gracias a esta prueba se descartó la posibilidad de utilizar la tarjeta Raspberry Pi 3 dado que no fue posible instalar las librerías antes mencionadas, quedándose estancada durante el proceso de instalación.

La segunda prueba consistió en la implementación de un controlador difuso como representante de los algoritmos de lógica difusa. Para ello se utilizó la librería eFLL en la tarjeta ESP, utilizando el ejercicio “simple sample”, el cual se encuentra como un ejemplo dentro de la propia librería. Y para realizar la prueba en la Raspberry se utilizó la librería Scikit-fuzzy, utilizando el ejercicio “Tipping Problem” que se explica en la documentación de la librería. La idea con esta primera prueba era que los algoritmos realizarán la misma tarea en entornos de programación y de ejecución diferentes. Como resultado del experimento se obtuvo un tiempo de respuesta de 13000 microsegundos en la tarjeta Raspberry y un tiempo de respuesta de 367 microsegundos en la ESP, siendo el tiempo de respuesta mayor en la Raspberry. Ambos algoritmos lograron llegar a la respuesta deseada.

La tercera prueba consistió en la implementación de algoritmos genéticos como representantes de la línea de computación evolutiva. Realizando la prueba se evidencio una dificultad en encontrar librerías que permitieran la programación de la tarjeta ESP, dentro de la búsqueda realizada sólo se encontró un artículo científico del 2010 que utilizaba Arduino como plataforma de desarrollo, el cual dejó de prestar soporte para la librería. Una solución alternativa podría ser el utilizar una tarjeta Arduino uno la cual, por la arquitectura interna del procesador podría utilizar la librería antes mencionada u otras librerías que se tuvieron que descartar durante el proceso de pruebas. Para realizar la prueba en la tarjeta Raspberry se utilizó la librería Genetic Algorithms, con el ejemplo “gene” que presenta la misma. El tiempo de respuesta obtenido en la prueba realizada en la tarjeta Raspberry fue de 0.7 segundos.

En este punto de las pruebas se hace evidente la necesidad de un cambio en el enfoque dado que el actual requería implementar un número indeterminado de pruebas, y requiriendo una delimitación más específica de los algoritmos, así como de su configuración interna. Igualmente, se evidencia la poca literatura con la que se cuenta para las tarjetas basadas en Arduino.

Probar la solución en el dominio original donde se identificó el problema: Luego de realizar las pruebas que tenían como objetivo proponer una comparativa que sirviera de guía para un usuario del patrón, se llegó a la conclusión de que no

se estaba siguiendo la estrategia correcta para su diseño, dado que el enfoque no era el apropiado para la investigación.

Segunda iteración del patrón

Observar: Al realizar la observación se evidencian problemas en el enfoque de la creación del patrón.

Identificar los problemas de investigación: El enfoque inicial buscaba delimitar el uso del patrón dependiendo de los resultados obtenidos al realizar pruebas de desempeño sin tener en cuenta el uso que se le dará a los algoritmos dentro de la solución y la practicidad que estos pueden tener en una determinada ubicación.

Desarrollar una solución al problema seleccionado: En esta segunda iteración se planteó realizar una selección de la ubicación dependiendo de la descripción del patrón, dada por el contexto y sus requisitos de uso, y de como este se adecua a las necesidades que tiene la aplicación IoT que se desea implementar.

Tomando como referente las características de un patrón de diseño definidas por Gamma et. al [1], junto con las características mencionadas por Bloom et. al [2], a continuación, se plantean las características del primer patrón arquitectónico:

a) Nombre: Patrón arquitectónico de procesamiento en el borde.

b) Intención: El patrón arquitectónico de procesamiento en el borde busca orientar al usuario en la implementación de una aplicación IoT que utiliza dispositivos en el borde para realizar el procesamiento y análisis de datos, utilizando algoritmos de inteligencia artificial.

c) Motivación: Dentro de un ecosistema de objetos inteligentes de la web de las cosas, existen distintos atributos de calidad a los cuales se puede enfocar el diseño de la arquitectura propuesta, dependiendo de este enfoque se puede tomar la decisión de realizar el procesamiento y análisis en una ubicación u otra de la arquitectura (Edge, Fog y Cloud). En el estudio titulado: “Distributing Intelligence among Cloud, Fog and Edge in Industrial Cyber-physical Systems”, Queiroz et. al [3], describen la distribución del procesamiento entre el Cloud, Fog y Edge, como se puede observar en la Figura 1. En este estudio, se presenta una distribución del procesamiento dependiendo de los atributos de calidad que se desean priorizar dentro del ecosistema. Y, además, plantea una discusión y un marco conceptual con el fin de mostrar la importancia que tiene el desarrollo de ecosistemas que utilicen las diferentes ubicaciones del procesamiento disponibles, por lo que se puede tomar como un estudio de partida que justifica la presente investigación.

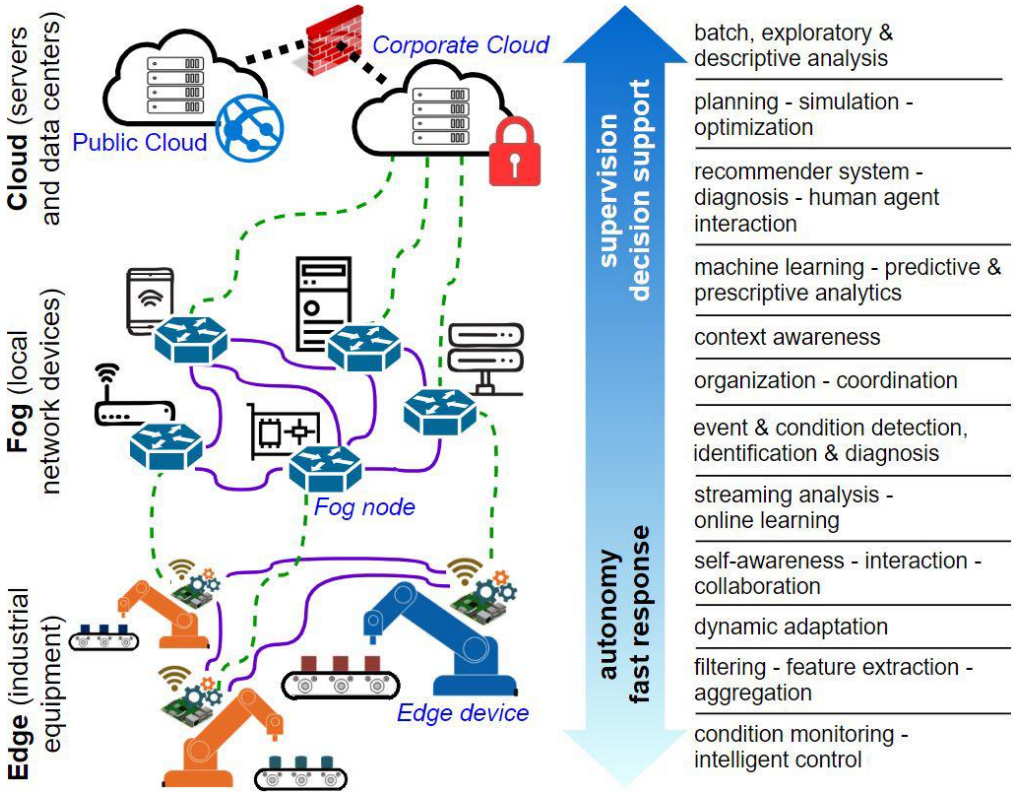

Figura 1. Distribución del procesamiento entre el Cloud Fog y Edge [3]

Por otra parte, dependiendo de la técnica de inteligencia computacional que se necesite implementar para dar solución a un determinado problema, en una aplicación específica, será necesario el uso de componentes hardware con unas capacidades de procesamiento y almacenamiento mínimas para poder desempeñarse en dicha tarea.

Cómo se concluye en la revisión sistemática de la literatura, al observar el estado del arte no se encontraron estudios que guíen al usuario en la selección de una ubicación en un ecosistema de objetos inteligentes utilizando algoritmos de inteligencia artificial.

d) Aplicabilidad (Contexto de uso): El patrón arquitectónico en el borde cuenta con una serie de requisitos previos que deben cumplirse para que el procesamiento en la ubicación sea viable. Los requisitos que debe tener en cuenta el usuario de este patrón son:

- Definir cual(es) enfoque(s) (categorías) de solución de problemas que abarcan la inteligencia artificial serán usados en la solución que está construyendo.

- El tiempo de respuesta al hacer una petición de procesamiento debe ser aceptable de acuerdo con el contexto de la aplicación (las necesidades del usuario que usara la solución y las capacidades del dispositivo).

- Tener debidamente establecida la tolerancia a errores con la cual contará la aplicación, así como la efectividad mínima tolerable.

- Las características hardware de los dispositivos disponibles deberán cumplir con los requerimientos mínimos para poder realizar el procesamiento y el análisis de los datos conforme la solución que se busca implementar.

- Contar con las herramientas necesarias para el desarrollo del procesamiento y análisis con inteligencia artificial.

- Los dispositivos deben contar con la capacidad para almacenar y acceder a todos los datos relevantes para realizar la solución requerida, ya sea porque los datos los captura el mismo dispositivo, o porque la adquisición de datos se realiza desde múltiples dispositivos a su alcance (en caso de implementar una red de dispositivos en el Edge).

- La aplicación IoT requiere de un nivel medio o bajo de seguridad para el tratamiento de la información. De lo contrario (nivel alto de seguridad) no se recomienda realizar el procesamiento y análisis en el borde de la red.

e) Estructura:

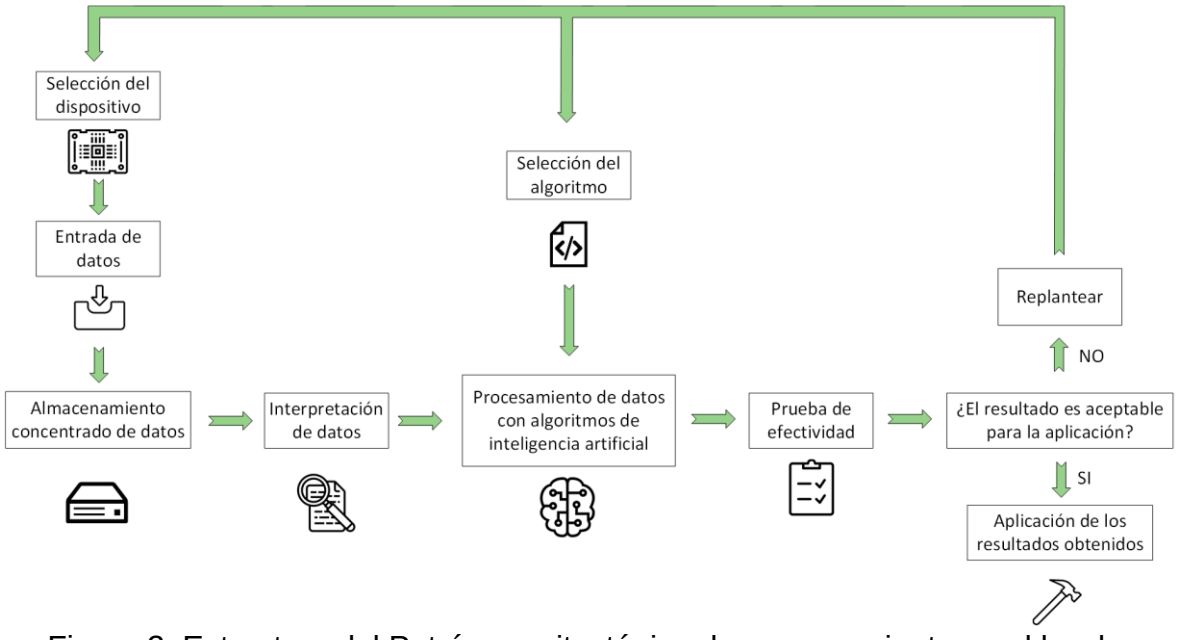

Figura 2. Estructura del Patrón arquitectónico de procesamiento en el borde. Imagen propia

f) Implementación:

El usuario debe seleccionar el dispositivo inteligente que realizara el procesamiento, en caso de ser más de un dispositivo se deberá seleccionar un dispositivo que concentre la información para almacenarla y realizar el procesamiento.

En el estudio titulado “A systematic literature review on hardware implementation of artificial intelligence algorithms”, Manar et. al [4], realizan una revisión sistemática de la literatura con el objetivo de comprender cuáles son las aplicaciones que integran algoritmos de inteligencia artificial dentro de dispositivos inteligentes existentes en la literatura. El estudio además busca identificar las técnicas de optimización de hardware utilizadas para implementar algoritmos de alta complejidad en dispositivos con baja capacidad de procesamiento como lo son los dispositivos en el borde. Dentro del estudio concluyen con algunas recomendaciones relacionadas con el hardware a utilizar dependiendo de los costos de adquisición de los dispositivos, tiempo de implementación y su flexibilidad para adaptarse a distintas aplicaciones. Las recomendaciones del estudio se muestran a continuación:

- Dado que el diseño ASIC (circuito integrado para aplicaciones específicas) no se puede modificar después de la fabricación, no es adecuado para áreas de aplicación donde el diseño debe actualizarse después de la implementación. ASIC se utiliza mejor cuando el objetivo es de baja potencia y área. Por el contrario, los FPGA (Field Programmable Gate Arrays) son flexibles para cambios y actualizaciones después de la implementación. Es importante mencionar que, la implementación de ASIC produce menos consumo de energía que la FPGA del mismo sistema.

- El tiempo de implementación de los FPGA se considera menor que el de los ASIC. Así mismo, los FPGA y ASIC brindan una mayor flexibilidad durante la fase de implementación en comparación con las GPU (Unidad de procesamiento gráfico).

- La implementación de GPU está restringida por el hardware subyacente existente, mientras que las FPGA y ASIC dependen completamente de la optimización del diseño realizada durante la fase de diseño.

- Las GPU están diseñadas con una jerarquía de memoria rápida con acceso directo a la memoria para resolver los problemas de ancho de banda. Esto permite tasas de transferencia más altas mientras toma menos tiempo. Las GPU se destacan en las operaciones de coma flotante, como las aplicaciones de procesamiento de señales e imágenes, gracias a los procesadores nativos de coma flotante.

- El costo de implementación tanto para GPU como para FPGA se considera medio en comparación con ASIC. Esto se debe al alto costo de fabricación de los ASIC.

Es recomendable observar las principales soluciones que se presentan en la literatura, las cuales se enfocan principalmente en seis categorías: Procesamiento de imágenes, procesamiento de datos, inteligencia ambiental, robótica, modelo de

indicador de recursos, y navegación mundial; siendo la categoría con mayor número de aplicaciones el procesamiento de imágenes [4].

Los enfoques de aplicación mencionados implementan algoritmos de inteligencia computacional relacionados a frameworks de trabajo, esta acción de características, aprendizaje profundo, aprendizaje no supervisado, clasificadores y modelos neuronales [4].

Probar la solución en el dominio original donde se identificó el problema

Al evaluar el patrón se observa qué es necesario mejorar su nivel de detalle con el fin de proporcionar una guía para el usuario más específica, y que permita obtener resultados satisfactorios.

Bibliografía

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

[2] G. Bloom, B. Alsulami, E. Nwafor, y 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.

[3] J. Queiroz, P. Leitão, J. Barbosa, y E. Oliveira, “Distributing Intelligence among Cloud, Fog and Edge in Industrial Cyber-physical Systems”, en *ICINCO*, 2019.

[4] M. A. Talib, S. Majzoub, Q. Nasir, y D. Jamal, “A systematic literature review on hardware implementation of artificial intelligence algorithms”, *J. Supercomput.*, vol. 77, núm. 2, pp. 1897–1938, feb. 2021, doi: 10.1007/S11227-020-03325-8/TABLES/15.