Universidad de Oriente
Núcleo de Nueva Esparta
Escuela de Ingeniería y Ciencias Aplicadas
Comisión de Trabajo de Grado

PROYECTO DE TRABAJO DE GRADO

Título Tentativo: Plataforma distribuida para el procesamiento de datos y comunicación de múltiples dispositivos o sistemas.

Modalidad: Tesis de Grado.

Autor: **Br.** Carlos Daniel Cañón Carrero.

Tutor Académico: **Lcdo.** Alfredo J. Arabia.

Autor.	Tutor académico.
Carlos D. Cañón.	Alfredo J. Arabia.
C.I. 21.306.802	C.I. 9.421.163

Guatamare, Junio 2018

Planteamiento del Problema

En el mundo actual, la información en tiempo real está siendo generada constantemente por distintas aplicaciones y dispositivos (de negocios, sociales, entre otras), esta información necesita formas simples y eficientes para ser enrutadas a múltiples receptores (Garg, 2013).

> "La computación en tiempo real juega un papel fundamental en nuestra sociedad, desde un incremento en la confianza de sistemas complejos, en parte o completamente en control computacional. Ejemplos que requieren computación en tiempo real incluye plantas de energía nuclear, sistemas de cambios de rieles, sistemas de automotores y aéreos, cosas como equipamiento médico, consumidores electrónicos, sistemas multimedia, sistemas de simulación de vuelos, realidad virtual, y juegos interactivos" (Buttazzo, 2011).

Adrian y Cassimally (2014), describen los principios de internet, cosas como el funcionamiento de los protocolos TCP y UDP. Adicionalmente, destacaron que algunos protocolos que se ejecutan sobre TCP, entre ellos los más destacados en Internet de las cosas (Internet of Things, IoT) fueron HTTP, MQTT y WebSockets; así mismo, comenta que la mayoría de dispositivos poseen un método propio de comunicación (protocolo), aunque destaca que en el futuro esto debería ser un estándar, pero por ahora lo más conveniente es utilizar uno propio y por ello es necesario facilitar el procesamiento de datos para protocolos de forma independiente.

En el Accounting Guide de Oracle® Communications (2018), se detalla la forma en que los Controladores de Borde de Sesión (Session Border Controllers, SBC) generan la información procesada, algo que llama la atención es la existencia de un método llamado llamado FTP Push, esto quiere decir que si realmente se quiere conectar más dispositivos, aparte los de Internet of Things (IoT), se debe contar con plataformas más moldeables porque en ocasiones no basta con sólo recibir mediante HTTP, WebSockets o MQTT, también hay diversos sistemas o dispositivos que usan métodos específicos para enviar, procesar y recibir información. Por ello se plantea el desarrollo de una nueva plataforma de procesamiento, comunicación y distribución de datos que facilite la tarea de recepción, procesamiento, homogeneización y distribución

de data. Cabe destacar que para facilitar la integración de nuevos sistemas se debe implementar múltiples métodos para manejar las sesiones entre los dispositivos o sistemas involucrados, además es necesario permitir una forma fácil de integrar nuevos protocolos al sistema.

De no implementar lo anteriormente expuesto, implicaría que para desarrollar plataformas de fácil despliegue, escalables y adaptables para intercomunicar sistemas, se necesitaría muchas horas para desarrollar rutinas que permitan dichas bondades, además de crear scripts para facilitar el despliegue y la conexión de las distintas tecnologías envueltas, así como también se requeriría muchas horas para hacer un sistema adaptable para procesar data de diversa índole, es decir procesar múltiples protocolos de forma independiente.

Un gran número de empresas y organizaciones administran sus sistemas en sus propios servidores, es decir, se ven imposibilitadas a usar servicios externos como Amazon IoT o Google Cloud. Es recomendable hacer un sistema Open Source, que sea portable y de fácil despliegue. Apache Kafka permite esto, pero aún siguen careciendo de características como permitir la integración fácil de nuevo protocolos, o manejo de sesiones.

Múltiples compañías podrían verse beneficiadas si se facilita el procesamiento de protocolos, un ejemplo serían las plataformas de rastreo, las empresas telefónicas u otras compañías que requieran recibir datos desde distintas fuentes, para luego procesarlos de forma independiente y homogeneizarlos; una vez hecho esto, podrían ser enrutados a otras aplicaciones o plataformas.

Objetivo General

Desarrollar una plataforma distribuida para el procesamiento de datos y comunicación de múltiples dispositivos o sistemas.

Objetivos Específicos

- Diseñar la arquitectura del sistema.
- Elaborar una plataforma distribuida, basada en microservicios para el procesamiento de datos y su intercomunicación.
- Automatizar rutinas para el despliegue de la aplicación.
- Ejecutar pruebas haciendo uso de pruebas unitarias.

Antecedentes

A continuación, se menciona algunos sistemas o plataformas con algunas similitudes al propuesto en este tema de investigación.

Como lo indican **Goodhope y Otros (2012),** en su publicación titulada **"Building LinkedIn's Real-time Activity Data Pipeline"**, Apache Kafka es una plataforma de streaming distribuida, es similar a un sistema de colas o sistema de mensajería empresarial ya que permite la publicación y subscripción a stream de registros. Adicionalmente permite almacenar registros de forma tolerante a fallos y permite procesamiento en tiempo real.

AWS IOT Core (2015). AWS IoT es una plataforma de nube administrada que permite a los dispositivos conectados interactuar con facilidad y seguridad con las aplicaciones en la nube y otros dispositivos. AWS IoT admite miles de millones de dispositivos y billones de mensajes, y es capaz de procesar y enrutar (entregar) dichos mensajes a puntos de enlace de AWS y a otros dispositivos de manera fiable y segura.

El SDK permite conectarse a los dispositivos únicamente mediantes los protocolos MQTT, HTTP 1.1 y WebSockets.

Lohrmann (2015) en su tesis doctoral titulada **“Massively Parallel Stream Processing with Latency Guarantees”** comenta sobre los motores de procesamiento de streams, dice que esto están basados en procesamiento por lote, además comenta que los mismos proveen facilidades para facilitar la distribución de data y a su vez facilitar el procesamiento paralelos y distribuidos. Comenta que esto es un reto muy grande para los desarrollarlos ya que en muchos sistemas, la latencia es algo fundamental, debido a que en ocasiones dicha latencia no puede superar unos pocos milisegundos.

Asad (2016) en su tesis de maestría **“Container-based IoT Sensor Node on Raspberry Pi and the Kubernetes Cluster Framework”** expone un sistema de IoT que se ejecuta sobre un cluster en kubernetes, que a su vez corre sobre dispositivos Raspberry Pi. adicionalmente usa kafka para entregar la información procesada a otros servicios además de proveer persistencia sobre hadoop.

Google Cloud IoT (2017). Google Cloud IoT es un conjunto de servicios completamente administrado e integrado que te permite conectar, administrar y consumir datos a gran escala, así como de forma fácil y segura, a partir de dispositivos repartidos por todo el mundo. También te ayuda a procesar, analizar y ver datos en tiempo real, implementar cambios operacionales y llevar a cabo las acciones que creas pertinentes. Google Cloud IoT provee comunicación mediante los protocolos HTTP y MQTT.

La investigación que se plantea, guarda relación con cada uno de los antecedentes mencionados debido a que todos tienen como fin recolectar y entregar datos, además de facilitar el procesamiento concurrente y distribuído.

El desarrollo brindará múltiples ventajas respecto a los anteriores, principalmente porque permitirá la integración de otros protocolos, dando así mayor facilidad para integrar nuevos dispositivos o sistemas, estos protocolos podrían ejecutarse sobre TCP/IP, UDP/IP o incluso diversos métodos como el FTP Push, mencionado anteriormente.

La escalabilidad es un tema importante, **Amazon IoT Core** y **Google Cloud IoT**, al pertenecer a un Cloud Computing (computación en la nube) tan poderoso, aseguran una escalabilidad gigantesca; **Apache Kafka** puede brindar gran escalabilidad, pero eso dependerá el número y las características de los servidores que lo alojen; por su parte el proyecto de **Asad** tiene como objetivo procesar información y entregarla a otras aplicaciones, es un excelente proyecto, pero tiene el objetivo de ejecutarse sobre Raspberries Pi y está absolutamente orientado al IoT. Por otra parte, el objetivo de este proyecto, en este aspecto, es brindar un servicio de gran escalabilidad, al igual que Apache Kafka, pero en una capa distinta de abstracción para facilitar su implementación y permitir la de forma simple el procesamiento e integración de nuevos protocolos.

Materiales y Métodos

Para la realización del proyecto, el investigador cuenta con los recursos materiales y financieros que acarrea la misma.

Materiales para la investigación y desarrollo:

- **Hardware** (Valor aproximado Bs.144.000.000,00)
    - **CPU:** i5-5200U 2.2 GHz; Dual-core
    - **RAM:** 16GB DDR3, 1600 MHz.
    - **Disco Duro:** 1TB Sata.
    - **DVD:** DVD RW.
    - **Tarjeta de Video:** Intel® HD 5500.

    - **Pantalla:** 15.6" HD (1366 x 768) resolution
- **Impresora:** Epson L355**.**
- **Software**
    - **Sistema Operativos:** Linux 64bits.
    - **Software de desarrollo:** distintas herramientas libres y open source (Rust-Lang, Scala, Java, apache ignite, akka, bash Linux).
    - **Entornos de desarrollo:** IntelliJ IDEA Community Edition, Visual Studio Code, Kate.
    - **Máquinas Virtuales y Emuladores:** Docker Community Edition.
- **Impresiones y Papelería** (Costo aproximado Bs.3.560.000,00)
    - Una (1) resma de papel tamaño carta.
    - Dos (2) lápices.
    - Un (1) Borrador de NATA.
    - Un (1) Sacapuntas.
    - Dos (2) CD's.
    - Empastado de tomo.
- **Hardware de implementación**
    - **CPU:** i5-5200U 2.2 GHz; Dual-core
    - **RAM:** 16GB DDR3, 1600 MHz.
    - **Disco Duro:** 1TB Sata.
    - **DVD:** DVD RW.
    - **Tarjeta de Video:** Intel® HD 5500.
    - **Pantalla:** 15.6" HD (1366 x 768) resolution

El investigador cuenta con todos los materiales y medios para el desarrollo del proyecto.

Metodología de la investigación

Según Hernández, Fernández y Baptista, (2010), la investigación científica se concibe como un conjunto de procesos sistemáticos y empíricos que se aplican al estudio de un fenómeno; es dinámica, cambiante y evolutiva.

El siguiente proyecto se considera como una investigación proyectiva.

Tipo de investigación: Según Hurtado (2010) se considera que una investigación proyectiva está orientada a elaborar una propuesta, un plan, un programa, un procedimiento, entre otros. También menciona que la investigación proyectiva se ocupa de cómo deberían ser las cosas, de tal modo que se puedan alcanzar sus objetivos y "funcionar" de forma correcta.

Tipo de diseño de la investigación: Hurtado (2010) comenta que aunque no existe una clasificación como tal, hay distintos tipos de proyectos y el que se implementará en este caso será del tipo tecnológico, ya que los proyectos tecnológicos incluyen diseño de software, maquinaria, artefactos, productos, sustancias químicas, edificaciones, materiales, alimentos (biotecnología), entre otros.

La presente investigación se considera de tipo proyecto factible, ya que cumple con la previa definición.

Las técnicas de recolección de datos serán la observación directa y análisis de documentos, la primera se basará en el análisis de los antecedentes planteados y vivencias en distintas empresas, mientras que en el análisis de documentos se empleará como fuente la revisión bibliográfica e internet.

A continuación se describirán las etapas propuestas por Hurtado para las investigaciones proyectivas.

- Determinar el enunciado holopráxico: Fase exploratoria.
- Desarrollar la justificación y plantear los objetivos: Fase descriptiva.
- Desarrollar la fundamentación noológica: Fase analítica, comparativa y explicativa.
- Revisar la factibilidad de la investigación: Fase predictiva.
- Precisar los lineamientos metodológicos: Fase proyectiva.
- Recoger los datos: Fase interactiva.
- Analizar, integrar y presentar los resultados: Fase confirmatoria.
- Evaluar el proceso: Fase evaluativa.

Metodología de desarrollo

Según Cortés (2017), Scrum es un framework ágil muy completo para el desarrollo de proyectos, está diseñado para entregar valor al cliente durante todo el desarrollo del proyecto y puede aplicarse a todo tipo de proyecto.

Según Lara (sin fecha), Scrum consta de las siguientes etapas:

1. Planeamiento del Sprint/Sprint Planning. Todos los involucrados del equipo se reúnen para planificar el Sprint,. Allí se asignan las tareas de cada miembro y estos mismos deben dar sus estimaciones de tiempo.
2. Reunión de Equipo de Scrum/Scrum team meeting. Reuniones diarias de máximo 15 mins y cada miembro debe responder las siguientes preguntas.
    a. ¿Qué hiciste ayer?
    b. ¿Qué tienes planeado hacer hoy?
    c. ¿Qué obstáculos encontraste en el camino?.
3. Refinamiento del Backlog/Backlog Refinement. Se revisa todos los eventos con el fin de esclarecer cualquier duda que pueda surgir por parte del equipo de desarrolladores.
4. Revisión del Sprint/Sprint Review. Todos se reúnen para mostrar los avances ejecutados durante el sprint.

5. Retrospectiva del Sprint/Retrospective. Reunión general donde para hablar sobre lo ocurrido durante el Sprint. Los puntos principales a tratar en esta reunión son:
    a. Qué se hizo mal durante el Sprint para poder mejorar el próximo
    b. Qué se hizo bien para seguir en la misma senda del éxito
    c. Qué inconvenientes se encontraron y no permitieron poder avanzar como se tenía planificado.

Cronograma de actividades

Basándose en los objetivos específicos, se contará con el siguiente cronograma de actividades.

1. **Recolección de información:** Para iniciar el desarrollo de la plataforma como requisito se debe hacer un análisis profundo de la composición y estructura de los sistemas de distribución de datos actuales, así como materiales, tecnologías y licencias para la elaboración del sistema. **Tiempo de ejecución aproximado dos (2) semanas.**

2. **Planificación y diseño del desarrollo:** Esto involucraría el diseño de la arquitectura así como la planificación de módulos y funciones principales incluidas en el sistema. Cabe destacar que al planificar las funciones anteriormente mencionadas, se debe tomar en cuenta los parámetros y las respuestas de las mismas, para así permitir el desarrollo de pruebas Test Driven Development (TDD). **Tiempo de ejecución aproximado tres (3) semanas.**

3. **Desarrollo de las pruebas:** Desarrollo guiado por pruebas de software, también conocido como Test Driven Development (TDD) es una práctica que involucra: Escribir las pruebas primero (Test First Development) y Refactorización (Refactoring). Primero se escriben las pruebas y luego se desarrolla y se verifica si falla. **Tiempo de desarrollo aproximado dos (2) semanas.**

4. **Elaboración de librerías:** La elaboración dichas librerías permitirán a la plataforma tomar ventajas de los antecedentes anteriormente mencionados, esas librerías incluirán la facilidad para integrar nuevos protocolos, rutinas para el procesamiento, recolección y distribución de data, entre otros. **Tiempo de desarrollo aproximado cuatro (4) semanas.**

5. **Desarrollo de plataforma:** Durante la ejecución de esta actividad se planea hacer uso de las librerías previamente desarrolladas, esto permitirá inte grar en una arquitectura de microservicios estas librerías y así completar la plataforma, dichas librerías proveerán funciones, mientras que la plataforma darán lógica y flujo a los datos**. Tiempo de desarrollo aproximado diez (10) semanas.**

6. **Implementación de docker:** A diferencia de los elementos anteriores del cronograma, esta sección es algo que se implementará durante toda la ejecución del proceso, las pruebas y todo los demás son algo que se realizarán desde docker. Luego de finalizar los puntos anteriores solo faltará hacer algunas pruebas desde esta herramienta. **Tiempo de desarrollo: durante todo el proyecto.**

7. **Ejecución de pruebas unitarias:** basados en las mismas pruebas del TDD, implementar rutinas para someter a prueba el funcionamiento del sistema. **Tiempo aproximado de ejecución, tres (3) semanas.**

8. **Liberación del código fuente:** Esta es una tarea que puede ejecutarse a lo largo del proyecto como al final, para ello solo es necesario subir el proyecto a un repositorio público. **Tiempo de desarrollo: durante todo el proyecto.**

Usando como referencia el cronograma anteriormente expuesto, a continuación se expondrá de forma gráfica, mostrando número de semanas ocupadas por el mismo.

Etapa \ Duración	**Junio**		**Julio**				**Agosto**				**Septiembre**				Octubre				
	4	5	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	5
Recolección de información																			
Planificación y diseño del desarrollo																			
Desarrollo de las pruebas																			
Elaboración de librerías																			
Desarrollo de plataforma																			
Implementación de docker																			
Ejecución de pruebas unitarias																			
Liberación del código fuente																			

REFERENCIAS

Hernández, R., Fernández, C., Baptista, M. (2010), Metodología de la Investigación Quinta Edición. Ciudad de México, México: McGRAW-HILL/INTERAMERICANA EDITORES, S.A. DE C.V.

Hurtado J. (2010), Metodología de la Investigación Guía la comprensión holística de la ciencia. Caracas, Venezuela: Ediciones Quirón.

Buttazzo, G. (Stankovic, J.). (2011), Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications. Nueva York, Estados Unidos: Springer Science+Business Media, LLC 2011.

Goodhope, K., Koshy, J., Kreps, J., Narkhede, N., Park, R., Rao, J., Ye, V. (2012), Building LinkedIn's Real-time Activity Data Pipeline. Recuperado de http://sites.computer.org/debull/A12june/pipeline.pdf

Garg, N. (2013), Learning Apache Kafka Second Edition. Birmingham, Reino Unido: Packt Publishing Ltd.

Adrian, M. y Cassimally H. (2014), Designing the Internet of Things. Sussex Occidental, Reino Unido: Wiley.

AWS IOT Core (2015). Amazon Web Services. Recuperado de https://aws.amazon.com/es/iot-core/

Lohrmann B. (2015), Massively Parallel Stream Processing with Latency Guarantees (Tesis doctoral). Technische Universität Berlin, Berlin, Alemania.

Asad, J. (2016), Container-based IoT Sensor Node on Raspberry Pi and the Kubernetes Cluster Framework (Tesis de maestria). Aalto University, Espoo, Finlandia.

GOOGLE CLOUD IOT (2017). Google Cloud Platform. Recuperado de https://cloud.google.com/solutions/iot/

Cortés, Y. L. (2017) , Qué es SCRUM y los roles en SCRUM. Platzi. Recuperado de https://platzi.com/blog/que-es-scrum-y-los-roles-en-scrum/

Lara, W. (sin fecha), Metodologia Scrum: Cómo funciona la metodología de trabajo Scrum. Platzi. Recuperado de https://platzi.com/blog/metodologia-scrum-fases/

Oracle® Communications (2018), Accounting Guide**.** Session Border Controller Documentation. Recuperado de https://docs.oracle.com/cd/E81287_01/doc/sbc_scz740_accounting.pdf

Universidad de Oriente
Núcleo de Nueva Esparta
Escuela de Ingeniería y Ciencias Aplicadas
Comisión de Trabajo de Grado

NOTA DE ENTREGA

Por medio de la presente, se entrega, ante la Comisión de Trabajo de Grado, cinco (5) ejemplares del Proyecto de Trabajo de Grado, modalidad Tesis de Grado titulado “Plataforma distribuida para el procesamiento de datos y comunicación de múltiples dispositivos o sistemas”, propuesto por el Br. Carlos Daniel Cañon Carrero, C.I. 21.306.802.

Recibido por: _____
Comisión de Trabajo de Grado

Fecha: Junio, 07 de 2018